INFO 523 Exercise

Exercise 1: Introduction to R Exercise

Table of contents

Part 1

Basics of R

Finding the R version

```
R.version
```

```
               _
platform       x86_64-pc-linux-gnu
arch           x86_64
os             linux-gnu
system         x86_64, linux-gnu
status
major          4
minor          3.1
year           2023
month          06
day            16
svn rev        84548
language       R
version.string R version 4.3.1 (2023-06-16)
nickname       Beagle Scouts
```

The code `R.version()` gives us the details of the platform, the kind of system on which our system operates (e.g., here it is a 64-bit operating system), the OS, the date, the version of R installed (4.3.1), and other general information about the environment where R has been installed.

Packages

Packages are important components of any programming language because they are like supporting pillars which makes our code run. There are several packages in R which will be used for various purposes.

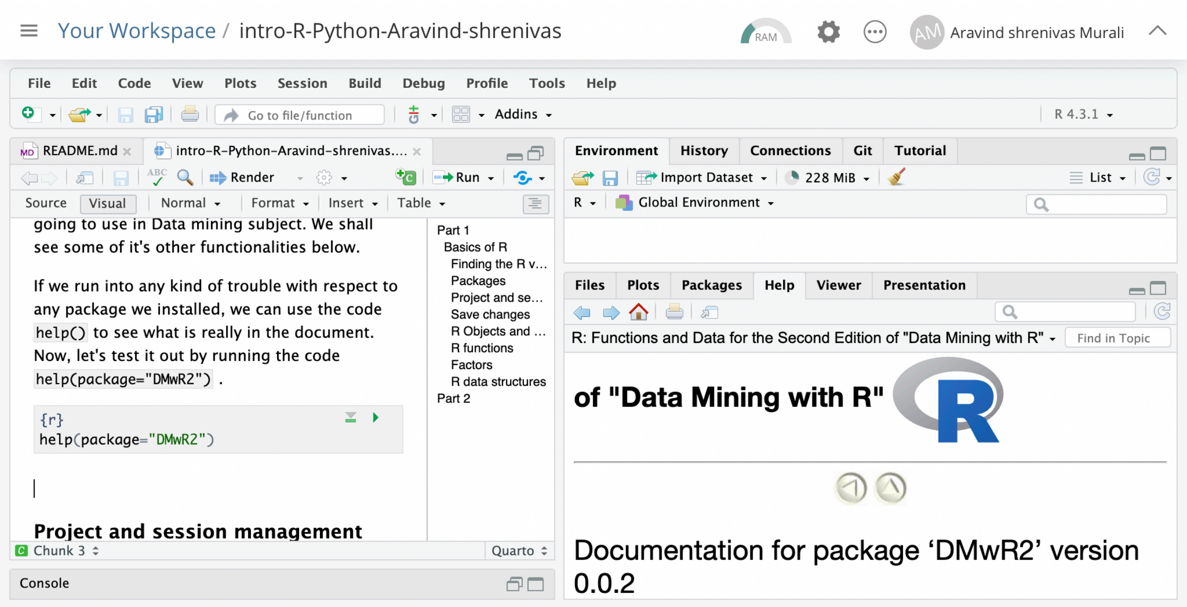
Let's install the package `DMwR2` . The syntax for this is `install.packages("DMwR2")` .

```
install.packages("DMwR2")
```

This is one of the main package which we are going to use in Data mining subject. We shall see some of it's other functionalities below.

If we run into any kind of trouble with respect to any package we installed, we can use the code `help()` to see what is really in the document. Now, let's test it out by running the code `help(package="DMwR2")` .

```
help(package="DMwR2")
```



going to use in Data mining subject. We shall see some of it's other functionalities below.

If we run into any kind of trouble with respect to any package we installed, we can use the code `help()` to see what is really in the document. Now, let's test it out by running the code `help(package="DMwR2")` .

```
{r}
help(package="DMwR2")
```

Project and session management

R: Functions and Data for the Second Edition of "Data Mining with R"

of "Data Mining with R"

Documentation for package 'DMwR2' version 0.0.2

Your Workspace / intro-R-Python-Aravind-shrenivas
RAM
AM Aravind shrenivas Murali
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
R 4.3.1
README.md
intro-R-Python-Aravind-shrenivas....
Render
Run
Source Visual
Normal Format Insert Table
Part 1
Basics of R
Finding the R v...
Packages
Project and se...
Save changes
R Objects and ...
R functions
Factors
R data structures
Part 2
Chunk 3
Quarto
Console
Environment History Connections Git Tutorial
Import Dataset
228 MiB
List
R
Global Environment
Files Plots Packages Help Viewer Presentation
Find in Topic
R

Figure 1: fig 1. help() Output

When I executed the `help(package="DMwR2")` command, the help menu which was on the side opened up which contains the compelte documentation for the package 'DMwR2'.

After installing of package, I need to use it. So, there are two ways by which I can access the package which I will list below.

1. There is a keyword called `library()` . When I want to use a function repeatedly, I can just load up the function to the temporary memory using this function for frequent use. For eg. let's say I want to use this 'DMwR2' package, the following code must be used.

```
library(DMwR2)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

Now, I can access any function or dataset associated with the package 'DMwR2' by using it's name directly. An example is given below.

```
# I will load an available dataset 'algae' directly by referencing it's name
data(algae)
algae
```

```
# A tibble: 200 x 18
   season size  speed   mxPH  mnO2    Cl    NO3   NH4  oPO4   PO4  Chla    a1
   <fct>  <fct> <fct>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 winter small medium  8       9.8  60.8  6.24  578   105   170   50      0  
 2 spring small medium  8.35   8    57.8  1.29  370   429.  559.   1.3    1.4
 3 autumn small medium  8.1   11.4  40.0  5.33  347.  126.  187.  15.6    3.3
 4 spring small medium  8.07   4.8  77.4  2.30   98.2  61.2 139.   1.4    3.1
 5 autumn small medium  8.06   9    55.4 10.4   234.   58.2  97.6 10.5    9.2
 6 winter small high    8.25  13.1  65.8  9.25  430    18.2  56.7 28.4   15.1
 7 summer small high    8.15  10.3  73.2  1.54  110    61.2 112.   3.2    2.4
 8 autumn small high    8.05  10.6  59.1  4.99  206.   44.7  77.4  6.9   18.2
 9 winter small medium  8.7    3.4  22.0  0.886 103.   36.3  71    5.54  25.4
10 winter small high    7.93   9.9   8    1.39    5.8  27.2  46.6  0.8   17  
# i 190 more rows
# i 6 more variables: a2 <dbl>, a3 <dbl>, a4 <dbl>, a5 <dbl>, a6 <dbl>,
#   a7 <dbl>
```

If I want to find the row number of entries which contains many NA data, I will use `manyNAs(algae)` .

```
manyNAs(algae)
```

```
[1]  62 199
```

From the above, we can infer that columns 62 and 199 contains many inaccurate data.

2. If I want to use a function only once or twice I can use the syntax `function/dataset` through the notation `package::functionname` .

```
DMwR2::algae
```

```
# A tibble: 200 x 18
   season size  speed   mxPH  mnO2    Cl    NO3   NH4  oPO4   PO4  Chla    a1
   <fct>  <fct> <fct>  <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 winter small medium  8      9.8  60.8  6.24  578   105   170    50     0
 2 spring small medium  8.35   8    57.8  1.29  370   429.  559.   1.3    1.4
 3 autumn small medium  8.1   11.4  40.0  5.33  347.  126.  187.  15.6    3.3
 4 spring small medium  8.07   4.8  77.4  2.30   98.2  61.2 139.   1.4    3.1
 5 autumn small medium  8.06   9    55.4 10.4   234.   58.2  97.6 10.5    9.2
 6 winter small high    8.25  13.1  65.8  9.25  430    18.2  56.7 28.4   15.1
 7 summer small high    8.15  10.3  73.2  1.54  110    61.2 112.   3.2    2.4
 8 autumn small high    8.05  10.6  59.1  4.99  206.   44.7  77.4   6.9   18.2
 9 winter small medium  8.7    3.4  22.0  0.886 103.   36.3 71     5.54  25.4
10 winter small high    7.93   9.9   8    1.39    5.8  27.2  46.6  0.8   17
# i 190 more rows
# i 6 more variables: a2 <dbl>, a3 <dbl>, a4 <dbl>, a5 <dbl>, a6 <dbl>,
#   a7 <dbl>
```

If I want to look into the installed packages in my system, I can use the code `library()` without passing any arguments inside it followed by `(.packages())` .

```
library()
(.packages())
```

```
[1] "DMwR2"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"
```

So, the above packages are loaded for my current session in my system.

To be more precise, I will consider `library()` to be a super set and the package which i want to check be a subset. Here `(.packages())` lists out all the available packages in the system.

The `datach` function removes a package installed. To demonstrate it, I will first install a package called ‘dbplyr’ and then I will remove it from my session using `detach` command.

```
install.packages("dbplyr") # I am installing the package
```

```
Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.3'
(as 'lib' is unspecified)
```

```
library(dbplyr)
(.packages()) # I am oading the package to the current session
```

```
[1] "dbplyr"    "DMwR2"     "stats"     "graphics"  "grDevices" "utils"
[7] "datasets"  "methods"   "base"
```

```
detach("package:dbplyr", unload=TRUE)
(.packages()) # I am detaching the package
```

```
[1] "DMwR2"     "stats"     "graphics"  "grDevices" "utils"     "datasets"
[7] "methods"   "base"
```

```
library(dplyr)# I am loading wanted library
```

```
Attaching package: 'dplyr'


The following objects are masked from 'package:stats':

    filter, lag


The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

In the above example, I have successfully detached the wrong package and added the right package.

I can also see the installed packages using the code `installed.packages()` .

```
installed.packages()
```

```
               Package          LibPath
base64enc      "base64enc"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
bit            "bit"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
bit64          "bit64"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
blob           "blob"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
```

```
bslib          "bslib"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
cachem         "cachem"         "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
cli            "cli"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
clipr          "clipr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
cpp11          "cpp11"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
crayon         "crayon"         "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
curl           "curl"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
DBI            "DBI"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
dbplyr         "dbplyr"         "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
digest         "digest"         "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
DMwR2          "DMwR2"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
dplyr          "dplyr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
ellipsis       "ellipsis"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
evaluate       "evaluate"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
fansi          "fansi"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
fastmap        "fastmap"        "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
fontawesome    "fontawesome"    "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
fs             "fs"             "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
generics       "generics"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
glue           "glue"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
highr          "highr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
hms            "hms"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
htmltools      "htmltools"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
jquerylib      "jquerylib"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
jsonlite       "jsonlite"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
knitr          "knitr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
lifecycle      "lifecycle"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
magrittr       "magrittr"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
memoise        "memoise"        "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
mime           "mime"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
palmerpenguins "palmerpenguins" "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
pillar         "pillar"         "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
pkgconfig      "pkgconfig"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
prettyunits    "prettyunits"    "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
progress       "progress"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
purrr          "purrr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
quantmod       "quantmod"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
R6             "R6"             "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
rappdirs       "rappdirs"       "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
readr          "readr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
rlang          "rlang"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
rmarkdown      "rmarkdown"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
sass           "sass"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
```

```
stringi        "stringi"        "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
stringr        "stringr"        "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
tibble         "tibble"         "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
tidyr          "tidyr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
tidyselect     "tidyselect"     "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
tinytex        "tinytex"        "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
TTR            "TTR"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
tzdb           "tzdb"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
utf8           "utf8"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
vctrs          "vctrs"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
vroom          "vroom"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
withr          "withr"          "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
xfun           "xfun"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
xts            "xts"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
yaml           "yaml"           "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
zoo            "zoo"            "/cloud/lib/x86_64-pc-linux-gnu-library/4.3"
base           "base"           "/opt/R/4.3.1/lib/R/library"
boot           "boot"           "/opt/R/4.3.1/lib/R/library"
class          "class"          "/opt/R/4.3.1/lib/R/library"
cluster        "cluster"        "/opt/R/4.3.1/lib/R/library"
codetools      "codetools"      "/opt/R/4.3.1/lib/R/library"
compiler       "compiler"       "/opt/R/4.3.1/lib/R/library"
datasets       "datasets"       "/opt/R/4.3.1/lib/R/library"
foreign        "foreign"        "/opt/R/4.3.1/lib/R/library"
graphics       "graphics"       "/opt/R/4.3.1/lib/R/library"
grDevices      "grDevices"      "/opt/R/4.3.1/lib/R/library"
grid           "grid"           "/opt/R/4.3.1/lib/R/library"
KernSmooth     "KernSmooth"     "/opt/R/4.3.1/lib/R/library"
lattice        "lattice"        "/opt/R/4.3.1/lib/R/library"
MASS           "MASS"           "/opt/R/4.3.1/lib/R/library"
Matrix         "Matrix"         "/opt/R/4.3.1/lib/R/library"
methods        "methods"        "/opt/R/4.3.1/lib/R/library"
mgcv           "mgcv"           "/opt/R/4.3.1/lib/R/library"
nlme           "nlme"           "/opt/R/4.3.1/lib/R/library"
nnet           "nnet"           "/opt/R/4.3.1/lib/R/library"
parallel       "parallel"       "/opt/R/4.3.1/lib/R/library"
rpart          "rpart"          "/opt/R/4.3.1/lib/R/library"
spatial        "spatial"        "/opt/R/4.3.1/lib/R/library"
splines        "splines"        "/opt/R/4.3.1/lib/R/library"
stats          "stats"          "/opt/R/4.3.1/lib/R/library"
stats4         "stats4"         "/opt/R/4.3.1/lib/R/library"
survival       "survival"       "/opt/R/4.3.1/lib/R/library"
tcltk          "tcltk"          "/opt/R/4.3.1/lib/R/library"
```

```
tools          "tools"          "/opt/R/4.3.1/lib/R/library"
utils          "utils"          "/opt/R/4.3.1/lib/R/library"
               Version    Priority
base64enc      "0.1-3"    NA
bit            "4.0.5"    NA
bit64          "4.0.5"    NA
blob           "1.2.4"    NA
bslib          "0.5.1"    NA
cachem         "1.0.8"    NA
cli            "3.6.1"    NA
clipr          "0.8.0"    NA
cpp11          "0.4.6"    NA
crayon         "1.5.2"    NA
curl           "5.0.2"    NA
DBI            "1.1.3"    NA
dbplyr         "2.3.3"    NA
digest         "0.6.33"   NA
DMwR2          "0.0.2"    NA
dplyr          "1.1.2"    NA
ellipsis       "0.3.2"    NA
evaluate       "0.21"     NA
fansi          "1.0.4"    NA
fastmap        "1.1.1"    NA
fontawesome    "0.5.2"    NA
fs             "1.6.3"    NA
generics       "0.1.3"    NA
glue           "1.6.2"    NA
highr          "0.10"     NA
hms            "1.1.3"    NA
htmltools      "0.5.6"    NA
jquerylib      "0.1.4"    NA
jsonlite       "1.8.7"    NA
knitr          "1.43"     NA
lifecycle      "1.0.3"    NA
magrittr       "2.0.3"    NA
memoise        "2.0.1"    NA
mime           "0.12"     NA
palmerpenguins "0.1.1"    NA
pillar         "1.9.0"    NA
pkgconfig      "2.0.3"    NA
prettyunits    "1.1.1"    NA
progress       "1.2.2"    NA
purrr          "1.0.2"    NA
```

```
quantmod       "0.4.25"   NA
R6             "2.5.1"    NA
rappdirs       "0.3.3"    NA
readr          "2.1.4"    NA
rlang          "1.1.1"    NA
rmarkdown      "2.24"     NA
sass           "0.4.7"    NA
stringi        "1.7.12"   NA
stringr        "1.5.0"    NA
tibble         "3.2.1"    NA
tidyr          "1.3.0"    NA
tidyselect     "1.2.0"    NA
tinytex        "0.46"     NA
TTR            "0.24.3"   NA
tzdb           "0.4.0"    NA
utf8           "1.2.3"    NA
vctrs          "0.6.3"    NA
vroom          "1.6.3"    NA
withr          "2.5.0"    NA
xfun           "0.40"     NA
xts            "0.13.1"   NA
yaml           "2.3.7"    NA
zoo            "1.8-12"   NA
base           "4.3.1"    "base"
boot           "1.3-28.1" "recommended"
class          "7.3-22"   "recommended"
cluster        "2.1.4"    "recommended"
codetools      "0.2-19"   "recommended"
compiler       "4.3.1"    "base"
datasets       "4.3.1"    "base"
foreign        "0.8-84"   "recommended"
graphics       "4.3.1"    "base"
grDevices      "4.3.1"    "base"
grid           "4.3.1"    "base"
KernSmooth     "2.23-21"  "recommended"
lattice        "0.21-8"   "recommended"
MASS           "7.3-60"   "recommended"
Matrix         "1.5-4.1"  "recommended"
methods        "4.3.1"    "base"
mgcv           "1.8-42"   "recommended"
nlme           "3.1-162"  "recommended"
nnet           "7.3-19"   "recommended"
parallel       "4.3.1"    "base"
```

```
rpart          "4.1.19"   "recommended"
spatial        "7.3-16"   "recommended"
splines        "4.3.1"    "base"
stats          "4.3.1"    "base"
stats4         "4.3.1"    "base"
survival       "3.5-5"    "recommended"
tcltk          "4.3.1"    "base"
tools          "4.3.1"    "base"
utils          "4.3.1"    "base"
               Depends
base64enc      "R (>= 2.9.0)"
bit            "R (>= 2.9.2)"
bit64          "R (>= 3.0.1), bit (>= 4.0.0), utils, methods, stats"
blob           NA
bslib          "R (>= 2.10)"
cachem         NA
cli            "R (>= 3.4)"
clipr          NA
cpp11          "R (>= 3.5.0)"
crayon         NA
curl           "R (>= 3.0.0)"
DBI            "methods, R (>= 3.0.0)"
dbplyr         "R (>= 3.1)"
digest         "R (>= 3.3.0)"
DMwR2          "R(>= 3.0), methods"
dplyr          "R (>= 3.5.0)"
ellipsis       "R (>= 3.2)"
evaluate       "R (>= 3.0.2)"
fansi          "R (>= 3.1.0)"
fastmap        NA
fontawesome    "R (>= 3.3.0)"
fs             "R (>= 3.4)"
generics       "R (>= 3.2)"
glue           "R (>= 3.4)"
highr          "R (>= 3.3.0)"
hms            NA
htmltools      "R (>= 2.14.1)"
jquerylib      NA
jsonlite       "methods"
knitr          "R (>= 3.3.0)"
lifecycle      "R (>= 3.4)"
magrittr       "R (>= 3.4.0)"
memoise        NA
```

```
mime           NA
palmerpenguins "R (>= 2.10)"
pillar         NA
pkgconfig      NA
prettyunits    NA
progress       NA
purrr          "R (>= 3.5.0)"
quantmod       "R (>= 3.2.0), xts(>= 0.9-0), zoo, TTR(>= 0.2), methods"
R6             "R (>= 3.0)"
rappdirs       "R (>= 3.2)"
readr          "R (>= 3.5)"
rlang          "R (>= 3.5.0)"
rmarkdown      "R (>= 3.0)"
sass           NA
stringi        "R (>= 3.1)"
stringr        "R (>= 3.3)"
tibble         "R (>= 3.4.0)"
tidyr          "R (>= 3.4.0)"
tidyselect     "R (>= 3.4)"
tinytex        NA
TTR            NA
tzdb           "R (>= 3.5.0)"
utf8           "R (>= 2.10)"
vctrs          "R (>= 3.5.0)"
vroom          "R (>= 3.4)"
withr          "R (>= 3.2.0)"
xfun           NA
xts            "R (>= 3.6.0), zoo (>= 1.7-12)"
yaml           NA
zoo            "R (>= 3.1.0), stats"
base           NA
boot           "R (>= 3.0.0), graphics, stats"
class          "R (>= 3.0.0), stats, utils"
cluster        "R (>= 3.5.0)"
codetools      "R (>= 2.1)"
compiler       NA
datasets       NA
foreign        "R (>= 4.0.0)"
graphics       NA
grDevices      NA
grid           NA
KernSmooth     "R (>= 2.5.0), stats"
lattice        "R (>= 4.0.0)"
```

```
MASS           "R (>= 4.0), grDevices, graphics, stats, utils"
Matrix         "R (>= 3.5.0), methods"
methods        NA
mgcv           "R (>= 3.6.0), nlme (>= 3.1-64)"
nlme           "R (>= 3.5.0)"
nnet           "R (>= 3.0.0), stats, utils"
parallel       NA
rpart          "R (>= 2.15.0), graphics, stats, grDevices"
spatial        "R (>= 3.0.0), graphics, stats, utils"
splines        NA
stats          NA
stats4         NA
survival       "R (>= 3.5.0)"
tcltk          NA
tools          NA
utils          NA
               Imports
base64enc      NA
bit            NA
bit64          NA
blob           "methods, rlang, vctrs (>= 0.2.1)"
bslib          "base64enc, cachem, grDevices, htmltools (>= 0.5.4), jquerylib\n(>= 0.1.3), js
cachem         "rlang, fastmap (>= 1.1.1)"
cli            "utils"
clipr          "utils"
cpp11          NA
crayon         "grDevices, methods, utils"
curl           NA
DBI            NA
dbplyr         "blob (>= 1.2.0), cli (>= 3.4.1), DBI (>= 1.0.0), dplyr (>=\n1.1.0), glue (>=
digest         "utils"
DMwR2          "xts (>= 0.9-7), zoo (>= 1.7-10), class (>= 7.3-14), rpart (>=\n4.1-10), quant
dplyr          "cli (>= 3.4.0), generics, glue (>= 1.3.2), lifecycle (>=\n1.0.3), magrittr (>
ellipsis       "rlang (>= 0.3.0)"
evaluate       "methods"
fansi          "grDevices, utils"
fastmap        NA
fontawesome    "rlang (>= 1.0.6), htmltools (>= 0.5.1.1)"
fs             "methods"
generics       "methods"
glue           "methods"
highr          "xfun (>= 0.18)"
hms            "lifecycle, methods, pkgconfig, rlang (>= 1.0.2), vctrs (>=\n0.3.8)"
```

```
htmltools      "utils, digest, grDevices, base64enc, rlang (>= 0.4.12),\nfastmap (>= 1.1.0),
jquerylib      "htmltools"
jsonlite       NA
knitr          "evaluate (>= 0.15), highr, methods, tools, xfun (>= 0.39),\nyaml (>= 2.1.19)"
lifecycle      "cli (>= 3.4.0), glue, rlang (>= 1.0.6)"
magrittr       NA
memoise        "rlang (>= 0.4.10), cachem"
mime           "tools"
palmerpenguins NA
pillar         "cli (>= 2.3.0), fansi, glue, lifecycle, rlang (>= 1.0.2), utf8\n(>= 1.1.0),
pkgconfig      "utils"
prettyunits    NA
progress       "hms, prettyunits, R6, crayon"
purrr          "cli (>= 3.6.1), lifecycle (>= 1.0.3), magrittr (>= 1.5.0),\nrlang (>= 1.1.1)
quantmod       "curl, jsonlite(>= 1.1)"
R6             NA
rappdirs       NA
readr          "cli (>= 3.2.0), clipr, crayon, hms (>= 0.4.1), lifecycle (>=\n0.2.0), method
rlang          "utils"
rmarkdown      "bslib (>= 0.2.5.1), evaluate (>= 0.13), fontawesome (>=\n0.5.0), htmltools (
sass           "fs (>= 1.2.4), rlang (>= 0.4.10), htmltools (>= 0.5.1), R6,\nrappdirs"
stringi        "tools, utils, stats"
stringr        "cli, glue (>= 1.6.1), lifecycle (>= 1.0.3), magrittr, rlang\n(>= 1.0.0), str
tibble         "fansi (>= 0.4.0), lifecycle (>= 1.0.0), magrittr, methods,\npillar (>= 1.8.1
tidyr          "cli (>= 3.4.1), dplyr (>= 1.0.10), glue, lifecycle (>= 1.0.3),\nmagrittr, pu
tidyselect     "cli (>= 3.3.0), glue (>= 1.3.0), lifecycle (>= 1.0.3), rlang\n(>= 1.0.4), vc
tinytex        "xfun (>= 0.29)"
TTR            "xts (>= 0.10-0), zoo, curl"
tzdb           NA
utf8           NA
vctrs          "cli (>= 3.4.0), glue, lifecycle (>= 1.0.3), rlang (>= 1.1.0)"
vroom          "bit64, cli (>= 3.2.0), crayon, glue, hms, lifecycle (>=\n1.0.3), methods, rl
withr          "graphics, grDevices, stats"
xfun           "stats, tools"
xts            "methods"
yaml           NA
zoo            "utils, graphics, grDevices, lattice (>= 0.20-27)"
base           NA
boot           NA
class          "MASS"
cluster        "graphics, grDevices, stats, utils"
codetools      NA
compiler       NA
```

```
datasets       NA
foreign        "methods, utils, stats"
graphics       "grDevices"
grDevices      NA
grid           "grDevices, utils"
KernSmooth     NA
lattice        "grid, grDevices, graphics, stats, utils"
MASS           "methods"
Matrix         "graphics, grid, lattice, stats, utils"
methods        "utils, stats"
mgcv           "methods, stats, graphics, Matrix, splines, utils"
nlme           "graphics, stats, utils, lattice"
nnet           NA
parallel       "tools, compiler"
rpart          NA
spatial        NA
splines        "graphics, stats"
stats          "utils, grDevices, graphics"
stats4         "graphics, methods, stats"
survival       "graphics, Matrix, methods, splines, stats, utils"
tcltk          "utils"
tools          NA
utils          NA
               LinkingTo
base64enc      NA
bit            NA
bit64          NA
blob           NA
bslib          NA
cachem         NA
cli            NA
clipr          NA
cpp11          NA
crayon         NA
curl           NA
DBI            NA
dbplyr         NA
digest         NA
DMwR2          NA
dplyr          NA
ellipsis       NA
evaluate       NA
fansi          NA
```

```
fastmap        NA
fontawesome    NA
fs             NA
generics       NA
glue           NA
highr          NA
hms            NA
htmltools      NA
jquerylib      NA
jsonlite       NA
knitr          NA
lifecycle      NA
magrittr       NA
memoise        NA
mime           NA
palmerpenguins NA
pillar         NA
pkgconfig      NA
prettyunits    NA
progress       NA
purrr          "cli"
quantmod       NA
R6             NA
rappdirs       NA
readr          "cpp11, tzdb (>= 0.1.1)"
rlang          NA
rmarkdown      NA
sass           NA
stringi        NA
stringr        NA
tibble         NA
tidyr          "cpp11 (>= 0.4.0)"
tidyselect     NA
tinytex        NA
TTR            "xts"
tzdb           "cpp11 (>= 0.4.2)"
utf8           NA
vctrs          NA
vroom          "cpp11 (>= 0.2.0), progress (>= 1.2.1), tzdb (>= 0.1.1)"
withr          NA
xfun           NA
xts            "zoo"
yaml           NA
```

```
zoo            NA
base           NA
boot           NA
class          NA
cluster        NA
codetools      NA
compiler       NA
datasets       NA
foreign        NA
graphics       NA
grDevices      NA
grid           NA
KernSmooth     NA
lattice        NA
MASS           NA
Matrix         NA
methods        NA
mgcv           NA
nlme           NA
nnet           NA
parallel       NA
rpart          NA
spatial        NA
splines        NA
stats          NA
stats4         NA
survival       NA
tcltk          NA
tools          NA
utils          NA
               Suggests
base64enc      NA
bit            "testthat (>= 0.11.0), roxygen2, knitr, rmarkdown,\nmicrobenchmark, bit64 (>=
bit64          NA
blob           "covr, crayon, pillar (>= 1.2.1), testthat"
bslib          "bsicons, curl, fontawesome, ggplot2, knitr, magrittr,\nrappdirs, rmarkdown (
cachem         "testthat"
cli            "callr, covr, crayon, digest, glue (>= 1.6.0), grDevices,\nhtmltools, htmlwidg
clipr          "covr, knitr, rmarkdown, rstudioapi (>= 0.5), testthat (>=\n2.0.0)"
cpp11          "bench, brio, callr, cli, covr, decor, desc, ggplot2, glue,\nknitr, lobstr, mo
crayon         "mockery, rstudioapi, testthat, withr"
curl           "spelling, testthat (>= 1.0.0), knitr, jsonlite, rmarkdown,\nmagrittr, httpuv
DBI            "blob, covr, DBItest, dbplyr, downlit, dplyr, glue, hms,\nknitr, magrittr, RMa
```

```
dbplyr          "bit64, covr, knitr, Lahman, nycflights13, odbc, RMariaDB (>=\n1.0.2), rmarkdo
digest          "tinytest, simplermarkdown"
DMwR2           NA
dplyr           "bench, broom, callr, covr, DBI, dbplyr (>= 2.2.1), ggplot2,\nknitr, Lahman, 
ellipsis        "covr, testthat"
evaluate        "covr, ggplot2, lattice, rlang, testthat (>= 3.0.0), withr"
fansi           "unitizer, knitr, rmarkdown"
fastmap         "testthat (>= 2.1.1)"
fontawesome     "covr, dplyr (>= 1.0.8), knitr (>= 1.31), testthat (>= 3.0.0),\nrsvg"
fs              "covr, crayon, knitr, pillar (>= 1.0.0), rmarkdown, spelling,\ntestthat (>= 3
generics        "covr, pkgload, testthat (>= 3.0.0), tibble, withr"
glue            "covr, crayon, DBI, dplyr, forcats, ggplot2, knitr, magrittr,\nmicrobenchmark
highr           "knitr, markdown, testit"
hms             "crayon, lubridate, pillar (>= 1.1.0), testthat (>= 3.0.0)"
htmltools       "markdown, testthat, withr, Cairo, ragg, shiny"
jquerylib       "testthat"
jsonlite        "httr, vctrs, testthat, knitr, rmarkdown, R.rsp, sf"
knitr           "bslib, codetools, DBI (>= 0.4-1), digest, formatR, gifski,\ngridSVG, htmlwidg
lifecycle       "covr, crayon, knitr, lintr, rmarkdown, testthat (>= 3.0.1),\ntibble, tidyvers
magrittr        "covr, knitr, rlang, rmarkdown, testthat"
memoise         "digest, aws.s3, covr, googleAuthR, googleCloudStorageR, httr,\ntestthat"
mime            NA
palmerpenguins "knitr, rmarkdown, tibble, ggplot2, dplyr, tidyr, recipes"
pillar          "bit64, DBI, debugme, DiagrammeR, dplyr, formattable, ggplot2,\nknitr, lubrida
pkgconfig       "covr, testthat, disposables (>= 1.0.3)"
prettyunits     "codetools, covr, testthat"
progress        "Rcpp, testthat, withr"
purrr           "covr, dplyr (>= 0.7.8), httr, knitr, lubridate, rmarkdown,\ntestthat (>= 3.0
quantmod        "DBI,RMySQL,RSQLite,timeSeries,xml2,downloader"
R6              "testthat, pryr"
rappdirs        "roxygen2, testthat (>= 3.0.0), covr, withr"
readr           "covr, curl, datasets, knitr, rmarkdown, spelling, stringi,\ntestthat (>= 3.1
rlang           "cli (>= 3.1.0), covr, crayon, fs, glue, knitr, magrittr,\nmethods, pillar, r
rmarkdown       "digest, dygraphs, fs, rsconnect, downlit (>= 0.4.0), katex\n(>= 1.4.0), sass
sass            "testthat, knitr, rmarkdown, withr, shiny, curl"
stringi         NA
stringr         "covr, htmltools, htmlwidgets, knitr, rmarkdown, testthat (>=\n3.0.0)"
tibble          "bench, bit64, blob, brio, callr, cli, covr, crayon (>=\n1.3.4), DiagrammeR, 
tidyr           "covr, data.table, knitr, readr, repurrrsive (>= 1.1.0),\nrmarkdown, testthat
tidyselect      "covr, crayon, dplyr, knitr, magrittr, rmarkdown, stringr,\ntestthat (>= 3.1.
tinytex         "testit, rstudioapi"
TTR             "RUnit"
tzdb            "covr, testthat (>= 3.0.0)"
```

```
utf8           "cli, covr, knitr, rlang, rmarkdown, testthat (>= 3.0.0),\nwithr"
vctrs          "bit64, covr, crayon, dplyr (>= 0.8.5), generics, knitr,\npillar (>= 1.4.4), p
vroom          "archive, bench (>= 1.1.0), covr, curl, dplyr, forcats, fs,\nggplot2, knitr, p
withr          "callr, covr, DBI, knitr, lattice, methods, rlang, rmarkdown\n(>= 2.12), RSQLi
xfun           "testit, parallel, codetools, rstudioapi, tinytex (>= 0.30),\nmime, markdown
xts            "timeSeries, timeDate, tseries, chron, tinytest"
yaml           "RUnit"
zoo            "AER, coda, chron, ggplot2 (>= 3.0.0), mondate, scales,\nstinepack, strucchang
base           "methods"
boot           "MASS, survival"
class          NA
cluster        "MASS, Matrix"
codetools      NA
compiler       NA
datasets       NA
foreign        NA
graphics       NA
grDevices      "KernSmooth"
grid           NA
KernSmooth     "MASS, carData"
lattice        "KernSmooth, MASS, latticeExtra, colorspace"
MASS           "lattice, nlme, nnet, survival"
Matrix         "MASS, expm"
methods        "codetools"
mgcv           "parallel, survival, MASS"
nlme           "Hmisc, MASS, SASmixed"
nnet           "MASS"
parallel       "methods"
rpart          "survival"
spatial        "MASS"
splines        "Matrix, methods"
stats          "MASS, Matrix, SuppDists, methods, stats4"
stats4         NA
survival       NA
tcltk          NA
tools          "codetools, methods, xml2, curl, commonmark, knitr, xfun, mathjaxr, V8"
utils          "methods, xml2, commonmark, knitr"
               Enhances
base64enc      "png"
bit            NA
bit64          NA
blob           NA
bslib          NA
```

```
cachem         NA
cli            NA
clipr          NA
cpp11          NA
crayon         NA
curl           NA
DBI            NA
dbplyr         NA
digest         NA
DMwR2          NA
dplyr          NA
ellipsis       NA
evaluate       NA
fansi          NA
fastmap        NA
fontawesome    NA
fs             NA
generics       NA
glue           NA
highr          NA
hms            NA
htmltools      "knitr"
jquerylib      NA
jsonlite       NA
knitr          NA
lifecycle      NA
magrittr       NA
memoise        NA
mime           NA
palmerpenguins NA
pillar         NA
pkgconfig      NA
prettyunits    NA
progress       NA
purrr          NA
quantmod       NA
R6             NA
rappdirs       NA
readr          NA
rlang          "winch"
rmarkdown      NA
sass           NA
stringi        NA
```

```
stringr        NA
tibble         NA
tidyr          NA
tidyselect     NA
tinytex        NA
TTR            "quantmod"
tzdb           NA
utf8           NA
vctrs          NA
vroom          NA
withr          NA
xfun           NA
xts            NA
yaml           NA
zoo            NA
base           NA
boot           NA
class          NA
cluster        NA
codetools      NA
compiler       NA
datasets       NA
foreign        NA
graphics       NA
grDevices      NA
grid           NA
KernSmooth     NA
lattice        "chron"
MASS           NA
Matrix         "MatrixModels, SparseM, graph, igraph, maptools, sfsmisc, sp,\nspdep"
methods        NA
mgcv           NA
nlme           NA
nnet           NA
parallel       "snow, Rmpi"
rpart          NA
spatial        NA
splines        NA
stats          NA
stats4         NA
survival       NA
tcltk          NA
tools          NA
```

```
utils          NA
               License                                  License_is_FOSS
base64enc      "GPL-2 | GPL-3"                          NA
bit            "GPL-2 | GPL-3"                          NA
bit64          "GPL-2 | GPL-3"                          NA
blob           "MIT + file LICENSE"                     NA
bslib          "MIT + file LICENSE"                     NA
cachem         "MIT + file LICENSE"                     NA
cli            "MIT + file LICENSE"                     NA
clipr          "GPL-3"                                  NA
cpp11          "MIT + file LICENSE"                     NA
crayon         "MIT + file LICENSE"                     NA
curl           "MIT + file LICENSE"                     NA
DBI            "LGPL (>= 2.1)"                          NA
dbplyr         "MIT + file LICENSE"                     NA
digest         "GPL (>= 2)"                             NA
DMwR2          "GPL (>= 2)"                             NA
dplyr          "MIT + file LICENSE"                     NA
ellipsis       "MIT + file LICENSE"                     NA
evaluate       "MIT + file LICENSE"                     NA
fansi          "GPL-2 | GPL-3"                          NA
fastmap        "MIT + file LICENSE"                     NA
fontawesome    "MIT + file LICENSE"                     NA
fs             "MIT + file LICENSE"                     NA
generics       "MIT + file LICENSE"                     NA
glue           "MIT + file LICENSE"                     NA
highr          "GPL"                                    NA
hms            "MIT + file LICENSE"                     NA
htmltools      "GPL (>= 2)"                             NA
jquerylib      "MIT + file LICENSE"                     NA
jsonlite       "MIT + file LICENSE"                     NA
knitr          "GPL"                                    NA
lifecycle      "MIT + file LICENSE"                     NA
magrittr       "MIT + file LICENSE"                     NA
memoise        "MIT + file LICENSE"                     NA
mime           "GPL"                                    NA
palmerpenguins "CC0"                                    NA
pillar         "MIT + file LICENSE"                     NA
pkgconfig      "MIT + file LICENSE"                     NA
prettyunits    "MIT + file LICENSE"                     NA
progress       "MIT + file LICENSE"                     NA
purrr          "MIT + file LICENSE"                     NA
quantmod       "GPL-3"                                  NA
```

```
R6             "MIT + file LICENSE"                     NA
rappdirs       "MIT + file LICENSE"                     NA
readr          "MIT + file LICENSE"                     NA
rlang          "MIT + file LICENSE"                     NA
rmarkdown      "GPL-3"                                  NA
sass           "MIT + file LICENSE"                     NA
stringi        "file LICENSE"                           "yes"
stringr        "MIT + file LICENSE"                     NA
tibble         "MIT + file LICENSE"                     NA
tidyr          "MIT + file LICENSE"                     NA
tidyselect     "MIT + file LICENSE"                     NA
tinytex        "MIT + file LICENSE"                     NA
TTR            "GPL (>= 2)"                             NA
tzdb           "MIT + file LICENSE"                     NA
utf8           "Apache License (== 2.0) | file LICENSE" NA
vctrs          "MIT + file LICENSE"                     NA
vroom          "MIT + file LICENSE"                     NA
withr          "MIT + file LICENSE"                     NA
xfun           "MIT + file LICENSE"                     NA
xts            "GPL (>= 2)"                             NA
yaml           "BSD_3_clause + file LICENSE"            NA
zoo            "GPL-2 | GPL-3"                          NA
base           "Part of R 4.3.1"                        NA
boot           "Unlimited"                              NA
class          "GPL-2 | GPL-3"                          NA
cluster        "GPL (>= 2)"                             NA
codetools      "GPL"                                    NA
compiler       "Part of R 4.3.1"                        NA
datasets       "Part of R 4.3.1"                        NA
foreign        "GPL (>= 2)"                             NA
graphics       "Part of R 4.3.1"                        NA
grDevices      "Part of R 4.3.1"                        NA
grid           "Part of R 4.3.1"                        NA
KernSmooth     "Unlimited"                              NA
lattice        "GPL (>= 2)"                             NA
MASS           "GPL-2 | GPL-3"                          NA
Matrix         "GPL (>= 2) | file LICENCE"              NA
methods        "Part of R 4.3.1"                        NA
mgcv           "GPL (>= 2)"                             NA
nlme           "GPL (>= 2)"                             NA
nnet           "GPL-2 | GPL-3"                          NA
parallel       "Part of R 4.3.1"                        NA
rpart          "GPL-2 | GPL-3"                          NA
```

```
spatial        "GPL-2 | GPL-3"                          NA
splines        "Part of R 4.3.1"                        NA
stats          "Part of R 4.3.1"                        NA
stats4         "Part of R 4.3.1"                        NA
survival       "LGPL (>= 2)"                            NA
tcltk          "Part of R 4.3.1"                        NA
tools          "Part of R 4.3.1"                        NA
utils          "Part of R 4.3.1"                        NA
               License_restricts_use OS_type MD5sum NeedsCompilation Built
base64enc      NA                    NA      NA     "yes"            "4.3.0"
bit            NA                    NA      NA     "yes"            "4.3.0"
bit64          NA                    NA      NA     "yes"            "4.3.0"
blob           NA                    NA      NA     "no"             "4.3.0"
bslib          NA                    NA      NA     "no"             "4.3.0"
cachem         NA                    NA      NA     "yes"            "4.3.0"
cli            NA                    NA      NA     "yes"            "4.3.0"
clipr          NA                    NA      NA     "no"             "4.3.0"
cpp11          NA                    NA      NA     "no"             "4.3.0"
crayon         NA                    NA      NA     "no"             "4.3.0"
curl           NA                    NA      NA     "yes"            "4.3.0"
DBI            NA                    NA      NA     "no"             "4.3.0"
dbplyr         NA                    NA      NA     "no"             "4.3.0"
digest         NA                    NA      NA     "yes"            "4.3.0"
DMwR2          NA                    NA      NA     "no"             "4.3.0"
dplyr          NA                    NA      NA     "yes"            "4.3.0"
ellipsis       NA                    NA      NA     "yes"            "4.3.0"
evaluate       NA                    NA      NA     "no"             "4.3.0"
fansi          NA                    NA      NA     "yes"            "4.3.0"
fastmap        NA                    NA      NA     "yes"            "4.3.0"
fontawesome    NA                    NA      NA     "no"             "4.3.0"
fs             NA                    NA      NA     "yes"            "4.3.0"
generics       NA                    NA      NA     "no"             "4.3.0"
glue           NA                    NA      NA     "yes"            "4.3.0"
highr          NA                    NA      NA     "no"             "4.3.0"
hms            NA                    NA      NA     "no"             "4.3.0"
htmltools      NA                    NA      NA     "yes"            "4.3.0"
jquerylib      NA                    NA      NA     "no"             "4.3.0"
jsonlite       NA                    NA      NA     "yes"            "4.3.0"
knitr          NA                    NA      NA     "no"             "4.3.0"
lifecycle      NA                    NA      NA     "no"             "4.3.0"
magrittr       NA                    NA      NA     "yes"            "4.3.0"
memoise        NA                    NA      NA     "no"             "4.3.0"
mime           NA                    NA      NA     "yes"            "4.3.0"
```

```
palmerpenguins NA                    NA      NA     "no"             "4.3.0"
pillar         NA                    NA      NA     "no"             "4.3.0"
pkgconfig      NA                    NA      NA     "no"             "4.3.0"
prettyunits    NA                    NA      NA     "no"             "4.3.0"
progress       NA                    NA      NA     "no"             "4.3.0"
purrr          NA                    NA      NA     "yes"            "4.3.0"
quantmod       NA                    NA      NA     "no"             "4.3.0"
R6             NA                    NA      NA     "no"             "4.3.0"
rappdirs       NA                    NA      NA     "yes"            "4.3.0"
readr          NA                    NA      NA     "yes"            "4.3.0"
rlang          NA                    NA      NA     "yes"            "4.3.0"
rmarkdown      NA                    NA      NA     "no"             "4.3.0"
sass           NA                    NA      NA     "yes"            "4.3.0"
stringi        NA                    NA      NA     "yes"            "4.3.0"
stringr        NA                    NA      NA     "no"             "4.3.0"
tibble         NA                    NA      NA     "yes"            "4.3.0"
tidyr          NA                    NA      NA     "yes"            "4.3.0"
tidyselect     NA                    NA      NA     "no"             "4.3.0"
tinytex        NA                    NA      NA     "no"             "4.3.0"
TTR            NA                    NA      NA     "yes"            "4.3.0"
tzdb           NA                    NA      NA     "yes"            "4.3.0"
utf8           NA                    NA      NA     "yes"            "4.3.0"
vctrs          NA                    NA      NA     "yes"            "4.3.0"
vroom          NA                    NA      NA     "yes"            "4.3.0"
withr          NA                    NA      NA     "no"             "4.3.0"
xfun           NA                    NA      NA     "yes"            "4.3.0"
xts            NA                    NA      NA     "yes"            "4.3.0"
yaml           NA                    NA      NA     "yes"            "4.3.0"
zoo            NA                    NA      NA     "yes"            "4.3.0"
base           NA                    NA      NA     NA               "4.3.1"
boot           NA                    NA      NA     "no"             "4.3.1"
class          NA                    NA      NA     "yes"            "4.3.1"
cluster        NA                    NA      NA     "yes"            "4.3.1"
codetools      NA                    NA      NA     "no"             "4.3.1"
compiler       NA                    NA      NA     NA               "4.3.1"
datasets       NA                    NA      NA     NA               "4.3.1"
foreign        NA                    NA      NA     "yes"            "4.3.1"
graphics       NA                    NA      NA     "yes"            "4.3.1"
grDevices      NA                    NA      NA     "yes"            "4.3.1"
grid           NA                    NA      NA     "yes"            "4.3.1"
KernSmooth     NA                    NA      NA     "yes"            "4.3.1"
lattice        NA                    NA      NA     "yes"            "4.3.1"
MASS           NA                    NA      NA     "yes"            "4.3.1"
```

```
Matrix         NA                    NA      NA     "yes"            "4.3.1"
methods        NA                    NA      NA     "yes"            "4.3.1"
mgcv           NA                    NA      NA     "yes"            "4.3.1"
nlme           NA                    NA      NA     "yes"            "4.3.1"
nnet           NA                    NA      NA     "yes"            "4.3.1"
parallel       NA                    NA      NA     "yes"            "4.3.1"
rpart          NA                    NA      NA     "yes"            "4.3.1"
spatial        NA                    NA      NA     "yes"            "4.3.1"
splines        NA                    NA      NA     "yes"            "4.3.1"
stats          NA                    NA      NA     "yes"            "4.3.1"
stats4         NA                    NA      NA     NA               "4.3.1"
survival       NA                    NA      NA     "yes"            "4.3.1"
tcltk          NA                    NA      NA     "yes"            "4.3.1"
tools          NA                    NA      NA     "yes"            "4.3.1"
utils          NA                    NA      NA     "yes"            "4.3.1"
```

To find if the installed packages have a newer version, I will use the code `old.packages()` .

```
old.packages()
```

```
           Package      LibPath                                      Installed
dplyr      "dplyr"      "/cloud/lib/x86_64-pc-linux-gnu-library/4.3" "1.1.2"
KernSmooth "KernSmooth" "/opt/R/4.3.1/lib/R/library"                 "2.23-21"
Matrix     "Matrix"     "/opt/R/4.3.1/lib/R/library"                 "1.5-4.1"
mgcv       "mgcv"       "/opt/R/4.3.1/lib/R/library"                 "1.8-42"
nlme       "nlme"       "/opt/R/4.3.1/lib/R/library"                 "3.1-162"
spatial    "spatial"    "/opt/R/4.3.1/lib/R/library"                 "7.3-16"
survival   "survival"   "/opt/R/4.3.1/lib/R/library"                 "3.5-5"
           Built   ReposVer
dplyr      "4.3.0" "1.1.3"
KernSmooth "4.3.1" "2.23-22"
Matrix     "4.3.1" "1.6-1"
mgcv       "4.3.1" "1.9-0"
nlme       "4.3.1" "3.1-163"
spatial    "4.3.1" "7.3-17"
survival   "4.3.1" "3.5-7"
           Repository
dplyr      "http://rspm/default/__linux__/focal/latest/src/contrib"
KernSmooth "http://rspm/default/__linux__/focal/latest/src/contrib"
Matrix     "http://rspm/default/__linux__/focal/latest/src/contrib"
mgcv       "http://rspm/default/__linux__/focal/latest/src/contrib"
```

```
nlme        "http://rspm/default/__linux__/focal/latest/src/contrib"
spatial     "http://rspm/default/__linux__/focal/latest/src/contrib"
survival    "http://rspm/default/__linux__/focal/latest/src/contrib"
```

Similarly I can update my packages using `update.packages()` .

```
update.packages()
```

Moreover, I can just type in the name of the function installed through a package by just entering the name. For example, `mean` is a function from base package.

```
mean
```

```
function (x, ...)
UseMethod("mean")
<bytecode: 0x563883e18f58>
<environment: namespace:base>
```

To find out more information about `mean` I can always use the code `help(mean)`. If there are two functions having the same name but belong to different packages, the code `package::functionname` shall be used to exactly specify the function and it's package.

```
help(mean)
```

Let's say if i want to do a neural networking project in R and I want to find the related packages, I can browse through the web using the code `RSiteSearch('neural networks')`

```
RSiteSearch('neural networks')
```

Project and session management

In this section, I will use posit cloud to demonstrate the management of project.

On the top menu bar, below `File` option there is a small `new file` option from where I can select the type of project I are going to work with. Some of them are R Script, R markdown, Quarto, etc. I can select the relevant option to me and start working on assignments or projects.

To resume the project, I can click on `File` and select `open file` . Now I have successfully resumed my project. This location is my project working directory, which means all my files will be stored here, especially .R and .Rdata files.

When finishing the project, it is always important to push all the changes to the github repository to prevent loss of data. After pushing the data into github, we can simply close the window.

There are some cool features in posit cloud. I can create a R script and run the entire script at once. and i can also render the code chunk by chunk.

It is a really good practice to save our changes frequently to prevent loss of data and for later use.

Save changes

To save the current workspace we should enter the following code. Note that, this code always saves the current workspace in .Rdata format.

```
save.image()
```

`getwd()` and `setwd()` will help us to get and set the working directory respctively.

```
getwd() #this will fetch the working directory

setwd("/enter/your/path/here") #I am setting a new directory here

getwd() #since I have set a new directory, that will show up when running this code
```

R Objects and Variables

Variables are like names to the memory location of a computer where it holds certain data and the data can be a simple number or a complex one.

```
hello1 <- 0.5 # I have assigned the variable hello1 with the value 0.5

hello1 #when i call this variable this should return 0.5
```

```
[1] 0.5
```

Use () to enclose a statement to have the returned values print directly:

```
(hello1 <- 0.5) #I have enclosed it with round brackets, hence this will print the values
```

```
[1] 0.5
```

Some examples:

```
x <- 5
y <- hello1 * x
y
```

```
[1] 2.5
```

```
z <-(y/2)^3
y
```

```
[1] 2.5
```

```
z
```

```
[1] 1.953125
```

All the declared variables continue to be alive until I delete it or exit posit cloud without saving. To list out the variables I can use the code `ls()` or `objects()`.

```
ls() #I am listing all the active objects in the current session
```

```
[1] "algae"           "algae.sols"      "has_annotations" "hello1"
[5] "test.algae"      "x"               "y"               "z"
```

```
objects() ##I am listing all the active objects in the current session
```

```
[1] "algae"           "algae.sols"      "has_annotations" "hello1"
[5] "test.algae"      "x"               "y"               "z"
```

Remove a variable to free memory space:

```
rm(hello1) #I am deleting hello1 variable from the working session

objects() #updated variables
```

```
[1] "algae"             "algae.sols"        "has_annotations" "test.algae"
[5] "x"                 "y"                 "z"
```

R functions

R functions are something which requires an input to give us an output by performing an operation. R has many functions and libraries that I can use in my program.

Some of the examples:

```
max(1, 5, 7, 12, -9) #this gives the maximum of input arguements
```

```
[1] 12
```

```
mean(1, 5, 7, 12, -4) #this finds the mean of the input arguements
```

```
[1] 1
```

```
max(sample(1:100, 50)) #this function generates 50 random numbers from 1 to 100 and finds
```

```
[1] 100
```

```
mean(sample(1:100, 30)) #this function generates 30 random numbers from 1 to 100 and finds
```

```
[1] 58.03333
```

```
help("sample") # this helps me to understand what sample does
```

```
set.seed(1) #the seed determines the starting point used in generating a sequence of pseud

#there is a function to remove the seed:rm(.Random.seed, envir=.GlobalEnv)

rnorm(1) #give me one number from a normal distribution
```

```
[1] -0.6264538
```

```
set.seed(5) #setting the seed to 5
rnorm(1) #give me one number from a normal distribution
```

```
[1] -0.8408555
```

`set.seed()` is basically used to produce the same output. Hence it is helpful in debugging of programs.

Now, I will create some custom functions. Before creating any function, it is important to check if the function exists. For that I will use the code `exists()` and pass in the name of the function as arguement. I want to create a function to find standard error of means `se` .

```
exists('se') #checking if func 'se' exists
```

```
[1] FALSE
```

So as seen above, the function doesn't exist. So let's create a new one.

```
se <- function(x){
  variance <- var(x)
  n <-length(x)
  return (sqrt(variance/n))
} #creating the function se
```

I have created the function. Let's verify,

```
exists('se') #checking if func 'se' exists
```

```
[1] TRUE
```

As we can see function ‘se’ is created successfully. Now let’s create another function with multiple arguements.

```
convMeters <- function (x, to="inch"){
  factor = switch(to, inch=39.3701, foot=3.28084, yard=1.09361, mile=0.000621371, NA)
  if(is.na(factor)) stop ("unknown target unit")
  else return (x*factor)
} #this function converts meters to inch, foot, yard and miles

convMeters(50, "foot") #testing the function
```

```
[1] 164.042
```

```
convMeters(40) #If no argument to is provided, the default value 'inch' is used
```

```
[1] 1574.804
```

```
convMeters(to="yard", 56.2) #arguements can also be given in other orders if they are name
```

```
[1] 61.46088
```

Factors

Factors are like a group of variables but they are limited. So, each factor is a category of unique variables. To create a factor we use the code `factor()`. factors are represented as internal numeric vectors.

Let’s create a factor which contains two categorical variables `f` and `m` .

```
g <-c('f', 'm', 'f', 'f', 'f', 'm', 'm', 'f')
g <- factor(g)
```

So, we have successfully created a factor with levels ‘f’ and ‘m’ . Another way of creating a factor is shown below:

```
other.g <-factor(c('m', 'm', 'm', 'm'), levels= c('f', 'm'))
other.g
```

```
[1] m m m m
Levels: f m
```

Let's now compare the above with the following variable:

```
other.g <-factor(c('m', 'm', 'm', 'm'))
other.g
```

```
[1] m m m m
Levels: m
```

The code correctly categorized the variables as level `m` .

The `table()` function helps us to categorize and summarize the data into a table. Let's see the demonstration below:

```
g <- factor(c('f', 'm', 'f', 'f', 'f', 'm', 'm', 'f'))
table(g) #we have created a table with factor of levels 'f' and 'm'
```

```
g
f m
5 3
```

I will add age factor to the table.

```
a <- factor(c('adult', 'juvenile','adult', 'juvenile','adult', 'juvenile','juvenile', 'juv
table(a, g) #I have successfully added factors 'adult' and 'juvenile'
```

```
          g
a          f m
  adult    3 0
  juvenile 2 3
```

By default, R assumes that both the factors belong to the same entity. Let's consider, in our dataset we have 3 female adult, 2 female juvenile, and 3 male juvenile.

```
a <- factor(c('adult', 'juvenile','adult', 'juvenile','adult', 'juvenile','juvenile'))
table(a, g)
```

```
#output: Error in table(a, g) : all arguments must have the same length
```

It says error because the factor is not of the same length of `g` . Now, I will create a new table of `a` which aligns with the length of `g` .

```
a <- factor(c('adult', 'juvenile','adult', 'juvenile','adult', 'juvenile','juvenile', 'juv
t <- table(a, g) #i have created a table which aligns with the length of g

t
```

```
          g
a          f m
  adult    3 0
  juvenile 2 3
```

Now let's find the marginal frequencies.

```
margin.table(t, 1)#1 refers to the first factor, a (age)
```

```
a
   adult juvenile
       3        5
```

```
margin.table(t, 2)#2 refers to the second factor, g
```

```
g
f m
5 3
```

Now I will find the relative frequencies with respect to each margin and the overall:

```
t #I am printing the table
```

```
          g
a          f m
  adult    3 0
  juvenile 2 3
```

```
prop.table(t, 1) #I am using the margin generated for the 1st factor a
```

```
          g
a             f   m
  adult    1.0 0.0
  juvenile 0.4 0.6
```

It says that juveniles are 40% female and 60% male and the adults are all males.

```
prop.table(t, 2) #I am using the margin generated for the 1st factor g
```

```
          g
a             f   m
  adult    0.6 0.0
  juvenile 0.4 1.0
```

```
prop.table(t) #overall
```

```
          g
a               f     m
  adult    0.375 0.000
  juvenile 0.250 0.375
```

Now, I will print the same output in percentages:

```
prop.table(t) * 100
```

```
          g
a              f    m
  adult    37.5  0.0
  juvenile 25.0 37.5
```

R data structures

Vectors

Vectors are one of the data objects. A number is a vector with single element. The elements in a vector should be of same data type.

```
v <- c(2, 5, 3, 4) #creating a vector
length(v)
```

```
[1] 4
```

```
mode(v) #this finds out the data type in the vector
```

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

```
v <- c(2, 5, 3, 4, 'me') #now, I will create a vector with strings and numeric variables

mode(v)
```

```
[1] "character"
```

```
v
```

```
[1] "2"  "5"  "3"  "4"  "me"
```

Now all the elements of the vector became character strings.

We can use `NA` to represent a special character. For eg.

```
v <- c(2, 5, 3, 4, NA)
mode(v)
```

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

```
v
```

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

As we can see above the `NA` did not affect the type of elements.

Now I will create a Boolean vector

```
b <- c(TRUE, FALSE, NA, TRUE)
mode(b) #the output will be 'logical' as this is boolean elements
```

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

```
b
```

```
[1]  TRUE FALSE    NA  TRUE
```

Elements in vectors are indexed from [1].

```
b[3] #printing the 3rd element
```

```
[1] NA
```

```
b[3] <- TRUE #replacing the third element and printing the vector
b
```

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

Vectors are elastic, so i can add data to any index

```
b[8] <- FALSE #I have added false to 8th position
b #printing the output
```

```
[1]  TRUE FALSE  TRUE  TRUE    NA    NA    NA FALSE
```

All the empty indexes are stored with missing value `NA` .

```
e <-vector() #creating an empty vector
mode(e)
```

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

```
e <- c()
mode(e)
```

```
[1] "NULL"
```

As we can see it says NULL which signifies that the vector is empty

```
length(e)
```

```
[1] 0
```

I will use vector elements to construct another vector

```
b2 <-c(b[1], b[3], b[5])
b2
```

```
[1] TRUE TRUE   NA
```

```
sqrt(v) #the square root of all elements in v
```

```
[1] 1.414214 2.236068 1.732051 2.000000       NA
```

Vector Arithmetic

Let's perform some arithmetic operations:

```
v1 <- c(3, 6, 9)
v2 <- c(1, 4, 8)
v1+v2 #addition
```

```
[1]  4 10 17
```

```
v1*v2 #dot product
```

```
[1]  3 24 72
```

```
v1-v2 #subtraction
```

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

```
v1/v2 #divsion
```

```
[1] 3.000 1.500 1.125
```

Recycling rule says that when performing arithmetic operations between two vectors, and if any one of the vector is of different length, the shorter vector will repeat it's elements starting from the index 1 of the same vector.

```
v3 <- c(1, 4)
v1+v3#the recycling rule makes v3 [1, 4, 1]
```

```
Warning in v1 + v3: longer object length is not a multiple of shorter object
length
```

```
[1]  4 10 10
```

A single number is also a vector

```
2*v1
```

```
[1]  6 12 18
```

Vector Summary

In this section we have seen that:

1. The elements in a vector are of same data types.
2. Vectors are elastic
3. Arithmetic operations of vectors
4. Recycling rule

```
mysum <- function (x){
  sum <- 0
  for(i in 1:length(x)){
    sum <- sum + x[i]
  }
  return (sum)
} #for loop

(mysum (c(1, 2, 3)))
```

```
[1] 6
```

In the above code the vector iterates inside the mysum and adds them, thus giving a output of 6.

Part 2

Easy ways to generate vectors

We can use () to print the result of a statement.

```
(x <-1:10) #printing nos from 1 to 10
```

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

```
(x <-10:1) #printing nos from 10 to 1
```

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

```
10:15-1 # the precedence of ':' is higher than arithmetic operators.
```

```
[1]  9 10 11 12 13 14
```

```
10:(15-1)
```

```
[1] 10 11 12 13 14
```

```
#we can use seq() to generate sequence
(seq(from=1, to=5, length=4)) # 4 values between 1 and 5 inclusive, even intervals/steps
```

```
[1] 1.000000 2.333333 3.666667 5.000000
```

```
(seq(length=10, from=-2, by=0.5)) #10 values, starting from 2, interval/step = 0.5
```

```
 [1] -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5
```

```
#rep(x, n): repeat x n times
(rep(5, 10))
```

```
 [1] 5 5 5 5 5 5 5 5 5 5
```

```
(rep("hi", 3)) #repeats hi 3 times
```

```
[1] "hi" "hi" "hi"
```

```
(rep(1:2, 3)) #print 1:2 3 times
```

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

```
(rep(TRUE:FALSE, 3)) #boolean
```

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

```
(rep(1:3, each=3)) #printing each of the elements 3 times
```

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

```
#gl() is for generating factor levels

gl(3, 5) #three levels, each repeat 5 times
```

```
 [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
Levels: 1 2 3
```

```
gl(2, 5, labels= c('female', 'male'))#two levels, each level repeat 5 times
```

```
 [1] female female female female female male   male   male   male   male  
Levels: female male
```

```
#first argument 2 says two levels.
#second argument 1 says repeat once
#third argment 20 says generate 20 values

gl(2, 1, 20, labels=c('female', 'male'))#10 alternating female and male pairs, a total of
```

```
 [1] female male   female male   female male   female male   female male  
[11] female male   female male   female male   female male   female male  
Levels: female male
```

```
#we can use factor() to convert number sequence to factor level labels

n <- rep(1:2, each=3)
(n <- factor(n,
             levels = c(1, 2),
             labels = c('female','male')
             ))
```

```
[1] female female female male   male   male  
Levels: female male
```

```
n
```

```
[1] female female female male   male   male  
Levels: female male
```

To generate random data according to some probability density functions: the functions has a general signature of `rfunc(n, par1, par2, …)`, where

1. `r` for random
2. `func` is the density function
3. `n` is the length of the data
4. `par1`, `par2`, … are the parameters

Question: Generate 10 values following a `normal distribution` with `mean = 10` and `standard deviation = 3`

```
#Answer

(rnorm(10, mean=10, sd=3))
```

```
 [1] 14.153078  6.233524 10.210428 15.134323  8.191276  8.583501  8.093886
 [8]  9.142679 10.414325 13.682891
```

```
#another example

(rt(10, df=5)) #generates a random sample of 10 values from a Student's t-distribution wit
```

```
 [1] -1.4504572  1.2754605 -1.4193131  0.7552904  1.3131134  1.0216921
 [7]  2.4727450 -0.8576598  0.6957255  0.8934850
```

Exercise :

1. Generate a random sample of `normally distributed` data of `size 100`, with a `mean of 20` and `standard deviation 4.`
2. Compute the standard error of means of the dataset.

```
#solution for 1
sample <- rnorm(100, mean=20, sd=4)
sample
```

```
  [1] 23.80630 15.96187 11.99811 12.95126 19.42957 26.20024 16.79031 19.70168
  [9] 27.58267 18.17372 22.24889 16.45197 18.15902 17.10269 19.72316 25.85299
 [17] 20.75090 24.08809 17.63266 19.55120 16.30019 23.01322 19.54956 19.74364
 [25] 20.93310 15.45367 23.41932 17.68652 21.98545 16.95977 18.63445 11.59068
 [33] 18.79319 14.91047 18.88134 19.18361 19.09754 21.38811 20.12947 21.65413
 [41] 19.37861 23.89394 20.48436 20.75669 17.74846 21.99366 13.03079 23.90212
 [49] 19.90367 22.70274 17.15876 29.54893 18.10627 19.69691 17.91264 23.70419
 [57] 15.75036 22.22814 23.60292 23.95978 21.53443 18.61366 17.83924 19.26978
 [65] 19.76280 12.01845 24.54125 22.70318 20.83393 19.76862 23.57525 19.08454
 [73] 12.13739 16.98596 25.12061 16.18838 26.48952 30.40057 20.55859 14.59712
 [81] 23.19572 13.78002 21.85488 20.20972 19.19187 24.68343 23.53938 14.72845
 [89] 13.42700 24.23700 21.16033 18.39987 24.97238 14.53436 14.23435 25.39420
 [97] 12.08589 15.03620 19.58384 22.93189
```

```
# solution for 2

# creating a function to calculate standard error of means
se <- function(x){
  variance <- var(x)
  n <-length(x)
  return (sqrt(variance/n))
}

#passing the arguement 'sample' to compute the result
se(sample)
```

```
[1] 0.4005357
```

Sub-setting

There are additional ways by which I can select the values from the vector.

```
#example

x <- c(0, -3, 4, -1, 45, 90, -5) #creating a vector

#select all elements that is greater than 0
(gtzero <- x[x>0])
```

```
[1]  4 45 90
```

We can use Boolean operators to select values.

```
#use of | (or), and & (and) operators

x <- c(0, -3, 4, -1, 45, 90, -5) #creating a vector

(x[x<=-2 | x>5]) #using 'or' operator
```

```
[1] -3 45 90 -5
```

```
(x[x>40 & x<100]) #using 'and' operator
```

```
[1] 45 90
```

We can use vector index to select values

```
x <- c(0, -3, 4, -1, 45, 90, -5) #creating a vector

(x[c(4, 6)])#select the 4th and 6th elements in the vector
```

```
[1] -1 90
```

```
(y<-c(4,6)) #similar example
```

```
[1] 4 6
```

```
(x[y]) #passing of vectors as index value arguements
```

```
[1] -1 90
```

```
(x[1:3]) #select the 1st to the 3rd elements in the vector
```

```
[1]  0 -3  4
```

We can use negative index to exclude elements

```r
x <- c(0, -3, 4, -1, 45, 90, -5)

(x[-1]) #select all but the first element
```

```
[1] -3  4 -1 45 90 -5
```

```r
(x[-c(4, 6)]) #excluding 4th and 6th element
```

```
[1]  0 -3  4 45 -5
```

```r
(x[-(1:3)]) #excluding first 3 elements
```

```
[1] -1 45 90 -5
```

Named elements

We can assign names to each value in a vector.

```r
x <- c(0, -3, 4, -1, 45, 90, -5) #creating a vector

names(x) <- c('s1', 's2', 's3', 's4', 's5', 's6', 's7') #assigning names

x
```

```
s1 s2 s3 s4 s5 s6 s7
 0 -3  4 -1 45 90 -5
```

```r
#another way of naming elements

(pH <- c(area1=4.5, area2=5.7, area3=9.8, mud=7.2))
```

```
area1 area2 area3   mud
  4.5   5.7   9.8   7.2
```

```
# we can use individual names to select the element

(pH['mud'])
```

```
mud
7.2
```

```
(pH[c('area1', 'mud')])
```

```
area1   mud
  4.5   7.2
```

We cannot exclude elements with it’s names.

```
(x[-s1]) #results in error
```

```
(x[-"s1"]) #results in error
```

```
(x[s1:s7]) #results in error
```

```
(x[c('s1':'s7')]) #results in error
```

```
#Empty index means to select all

(pH[])
```

```
area1 area2 area3   mud
  4.5   5.7   9.8   7.2
```

```
pH
```

```
area1 area2 area3   mud
  4.5   5.7   9.8   7.2
```

To reset a vector to ‘0’ we use,

```
pH[] <- 0
pH #assigning 0 to pH
```

```
area1 area2 area3   mud 
    0     0     0     0 
```

```
pH<- 0
pH #same as above
```

```
[1] 0
```

More R-Data structures

Matrices and Arrays

Arrays and Matrices are long vectors categorized by dimensions. Moreover, Arrays can be of multiple dimension, whereas Matrices are two dimensional. They both hold the same type of value.

Matrices

```
#To create a matrix:

m <- c(45, 23, 66, 77, 33, 44, 56, 12, 78, 23) #creating a vector

is.vector(m) #checking if this is a vector
```

```
[1] TRUE
```

```
is.matrix(m) #checking if this is a matrix
```

```
[1] FALSE
```

```
is.array(m) #checking if this is an array
```

```
[1] FALSE
```

```r
#now 'organize' the vector as a matrix

dim(m) <-c(2, 5)#make the vector a 2 by 5 matrix, 2x5 must = lenght of the vector

m
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   23   77   44   12   23
```

```r
#re-checking

(is.vector(m))
```

```
[1] FALSE
```

```r
(is.matrix(m))
```

```
[1] TRUE
```

```r
(is.array(m))
```

```
[1] TRUE
```

The elements are put in matrix in columns by default. If we want to use in rows, we should use the code `byrow=TRUE`.

```r
#example

(m <- matrix(c(45, 23, 66, 77, 33, 44, 56, 12, 78, 23), 2, 5, byrow = TRUE))
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   45   23   66   77   33
[2,]   44   56   12   78   23
```

Exercise:

First columns hold age data for a group of students `11, 11, 12, 13, 14, 9, 8`, and second columns hold grades `5, 5, 6, 7, 8, 4, 3`.

```
#solution

studentsAndGrades <-matrix(c(11, 11, 12, 13, 14, 9, 8, 5, 5, 6, 7, 8, 4, 3), 7, 2) #creati

studentsAndGrades
```

```
     [,1] [,2]
[1,]   11    5
[2,]   11    5
[3,]   12    6
[4,]   13    7
[5,]   14    8
[6,]    9    4
[7,]    8    3
```

Same as vectors, we can access matrix by their position index.

```
#creating a matrix

m <- c(45, 23, 66, 77, 33, 44, 56, 12, 78, 23)
#then 'organize' the vector as a matrix

dim(m) <- c(2, 5)#make the vector a 2 by 5 matrix, 2x5 must = lenght of the vector

m
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   23   77   44   12   23
```

```
m[2, 3]#the element at row 2 and column 3
```

```
[1] 44
```

Similarly, we can use sub-setting for matrix also. The result will be a value (a value is a vector), a vector, or a matrix.

```
(s<- m[2, 1]) # select one value
```

```
[1] 23
```

```
(m<- m [c(1,2), -c(3, 5)]) #select 1st row and 1st, 2nd, and 4th columns, result is a vect
```

```
     [,1] [,2] [,3]
[1,]   45   66   56
[2,]   23   77   12
```

```
(m [1, ]) #select complete row or column: 1st row, result is a vector
```

```
[1] 45 66 56
```

```
(v<-m [, 1]) # 1st column, result is a vector
```

```
[1] 45 23
```

```
#performing checks to verify

is.vector(m)
```

```
[1] FALSE
```

```
is.matrix(m)
```

```
[1] TRUE
```

```
is.vector(s)
```

```
[1] TRUE
```

```
is.vector(v)
```

```
[1] TRUE
```

```
is.matrix(v)
```

```
[1] FALSE
```

```
#Use drop = FALSE to keep the results as a matrix

m <- matrix(c(45, 23, 66, 77, 33, 44, 56, 12, 78, 23), 2, 5)
(m<-m [, 2, drop = FALSE])
```

```
     [,1]
[1,]   66
[2,]   77
```

```
is.matrix(m)
```

```
[1] TRUE
```

```
is.vector(m)
```

```
[1] FALSE
```

If we want to join together two or more vectors or matrices, by column, or by row, respectively, we can use the code `cbind()` and `rbind()`.

```
#example

cbind (c(1,2,3), c(4, 5, 6)) #joining columns
```

```
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

```
rbind (c(1,2,3), c(4, 5, 6)) #joining rows
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

```
m <- matrix(c(45, 23, 66, 77, 33, 44, 56, 12, 78, 23), 2, 5)

(a <- rbind (c(1,2,3,4,5), m)) #joining a to m as rows
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]   45   66   33   56   78
[3,]   23   77   44   12   23
```

```
is.array(a)
```

```
[1] TRUE
```

```
is.matrix(a)
```

```
[1] TRUE
```

Exercise:

What will `m1-m4` look like?

```
#solution
m1 <- matrix(rep(10, 9), 3, 3)
m1
```

```
     [,1] [,2] [,3]
[1,]   10   10   10
[2,]   10   10   10
[3,]   10   10   10
```

```
m2 <- cbind (c(1,2,3), c(4, 5, 6))
m2
```

```
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

```
m3 <- cbind (m1[,1], m2[2,])
```

```
Warning in cbind(m1[, 1], m2[2, ]): number of rows of result is not a multiple
of vector length (arg 2)
```

```
m3
```

```
     [,1] [,2]
[1,]   10    2
[2,]   10    5
[3,]   10    2
```

```
m4 <- cbind (m1[,1], m2[,2])
m4
```

```
     [,1] [,2]
[1,]   10    4
[2,]   10    5
[3,]   10    6
```

Since m3 number of rows of result is not a multiple of vector length m2, it is not possible to bind them.

Named rows and columns

```
#we can name elements in matrix
```

```
sales <- matrix(c(10, 30, 40, 50, 43, 56, 21, 30), 2, 4, byrow=TRUE)
colnames(sales) <- c('1qrt', '2qrt', '3qrt', '4qrt')
rownames(sales) <- c('store1', 'store2')
sales
```

```
       1qrt 2qrt 3qrt 4qrt
store1   10   30   40   50
store2   43   56   21   30
```

Exercise:

1. Find `store1` `1qrt` sale.
2. List `store2`'s 1st and 4th quarter sales

```
#solution

sales['store1', '1qrt']
```

```
[1] 10
```

```
sales['store2', c('1qrt', '4qrt')]
```

```
1qrt 4qrt
  43   30
```

Arrays

Arrays and Matrices are almost same but arrays can have more than 2 dimensions.

```
#an example for 3-D array

a <- array(1:48, dim= c(4, 3, 2))
a
```

```
, , 1

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

, , 2

     [,1] [,2] [,3]
[1,]   13   17   21
[2,]   14   18   22
[3,]   15   19   23
[4,]   16   20   24
```

If we select array elements using indexes, results may be a value, a vector, a matrix or an array, depending on the use of the code `drop=FALSE` .

```
a [1, 3, 2] #a[1, 3, 2] refers to the element in the first dimension (, , 1 ), third row,
```

```
[1] 21
```

```
a [1, , 2]
```

```
[1] 13 17 21
```

```
a [1, , 2, drop=FALSE] #the dimensions are preserved since we have set drop=FALSE
```

```
, , 1

     [,1] [,2] [,3]
[1,]   13   17   21
```

```
a [4, 3, ]
```

```
[1] 12 24
```

```
a [c(2, 3), , -2]
```

```
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    3    7   11
```

Now we will assign names to dimensions of an array.

The code `[ [] ]` selects one dimension:

```
dimnames(a)[[1]] <-c("1qrt", "2qrt", "3qrt", "4qrt")
dimnames(a)[[2]] <-c("store1", "store2", "store3")
dimnames(a)[[3]] <-c("2017", "2018")
a
```

```
, , 2017

     store1 store2 store3
1qrt      1      5      9
2qrt      2      6     10
3qrt      3      7     11
4qrt      4      8     12

, , 2018

     store1 store2 store3
1qrt     13     17     21
2qrt     14     18     22
3qrt     15     19     23
4qrt     16     20     24
```

Alternatively, use `list()` to specify names:

```
ar <- array(data     = 1:27,
            dim      = c(3, 3, 3),
            dimnames = list(c("a", "b", "c"),
ar
```

```
, , g

  d e f
a 1 4 7
b 2 5 8
c 3 6 9

, , h

   d  e  f
a 10 13 16
b 11 14 17
c 12 15 18

, , i

   d  e  f
a 19 22 25
b 20 23 26
c 21 24 27
```

Split array into matrices

Now we will perform arithmetic operations on matrices, keeping in mind the recycling rule. Recycling rule says that when performing arithmetic operations between two vectors, and if any one of the vector is of different length, the shorter vector will repeat it's elements starting from the index 1 of the same vector.

```
matrix1 <- ar[,,'g'] #assigning 'g' to matrix1
matrix1
```

```
  d e f
a 1 4 7
b 2 5 8
c 3 6 9
```

```
matrix2 <- ar[,,'h'] ##assigning 'h' to matrix1
matrix2
```

```
   d  e  f
```

```
a 10 13 16
b 11 14 17
c 12 15 18
```

```
sum <-matrix1 + matrix2 #addition
sum
```

```
   d  e  f
a 11 17 23
b 13 19 25
c 15 21 27
```

```
matrix1*3 #matrix multiplication by scalar
```

```
  d  e  f
a 3 12 21
b 6 15 24
c 9 18 27
```

A matrix is just a long vector organized into dimensions, note the recycling rules apply:

```
matrix1
```

```
  d e f
a 1 4 7
b 2 5 8
c 3 6 9
```

```
matrix1*c(2, 3)
```

```
Warning in matrix1 * c(2, 3): longer object length is not a multiple of shorter
object length
```

```
  d  e  f
a 2 12 14
b 6 10 24
c 6 18 18
```

```
matrix1*c(2,3,2,3,2,3,2,3,2)
```

```
  d  e  f
a 2 12 14
b 6 10 24
c 6 18 18
```

```
matrix1*c(1, 2, 3)
```

```
  d  e  f
a 1  4  7
b 4 10 16
c 9 18 27
```

```
matrix1/c(1, 2, 3)
```

```
  d   e f
a 1 4.0 7
b 1 2.5 4
c 1 2.0 3
```

```
matrix1/c(1, 2, 3, 1, 2, 3, 1, 2, 3)
```

```
  d   e f
a 1 4.0 7
b 1 2.5 4
c 1 2.0 3
```

Lists

Lists are vectors as well, but they are'recursive' (as opposed to the 'atomic' vectors), which means they can hold other lists, which means a list can hold data of multiple sorts. Lists are made up of an ordered collection of items called as components. The list components do not have to have the same type. List components are always numbered (with an index) and may also be given a name.

We will use `list$component_name` to access a component in a *list* can not be used on atomic vectors.

```
mylist <- list(stud.id=34453,
               stud.name="John",
               stud.marks= c(13, 3, 12, 15, 19)
               ) #creating a list

mylist$stud.id #printing student id
```

```
[1] 34453
```

```
mylist[1] #accessing with index
```

```
$stud.id
[1] 34453
```

```
mylist[[1]] #[[]] will print the value directly
```

```
[1] 34453
```

```
mylist["stud.id"]
```

```
$stud.id
[1] 34453
```

```
handle <- "stud.id"
mylist[handle] #assigning the student id to handle and retrieving it back
```

```
$stud.id
[1] 34453
```

```
mylist[["stud.id"]]
```

```
[1] 34453
```

Subset with `[`

The subset can be extracted using both indices and names. To use names, an object must contain a name type attribute such as names, rownames, colnames, and so on.

Negative numbers can be used to signify exclusion.

Variables that are not quoted are interpolated within the brackets.

```r
#example

mylist[1]
```

```
$stud.id
[1] 34453
```

Extract one item with `[[`

The double square brackets are used to extract one element from a potentially large number of them. For vectors, a single value is returned; for data frames, a column vector is returned; and for lists, one element is returned.

I may only return one item. The end result is not (necessarily) the same . The dimension will be the dimension of the single item, which may or may not be 1. And, as previously stated, either names or indexes can be utilised. Variables are interpolated.

```r
#example

mylist[[1]]
```

```
[1] 34453
```

Interact with `$`

`$` is a particular case of `[[` that allows one to access a single item by name (but not for atomic vectors). Integer indexing are not permitted.

The name will not be interpolated, and only one item will be returned. If the name contains special characters, it must be surrounded by backticks: `"`

```
mylist <- list(stud.id=34453,
               stud.name="John",
               stud.marks= c(13, 3, 12, 15, 19)
               )
mylist$stud.marks
```

```
[1] 13  3 12 15 19
```

```
mylist$stud.marks[2]
```

```
[1] 3
```

Change names:

```
names(mylist) #printing the existing names
```

```
[1] "stud.id"    "stud.name"  "stud.marks"
```

```
names(mylist) <- c('id','name','marks') #assigning new names

names(mylist)
```

```
[1] "id"    "name"  "marks"
```

```
mylist
```

```
$id
[1] 34453

$name
[1] "John"

$marks
[1] 13  3 12 15 19
```

Add new component:

```r
mylist$parents.names <- c('Ana', "Mike")
mylist
```

```
$id
[1] 34453

$name
[1] "John"

$marks
[1] 13  3 12 15 19

$parents.names
[1] "Ana"  "Mike"
```

One should use `c()` to concatenate two lists:

```r
newlist <- list(age=19, sex="male"); #declaring a newlist

expandedlist <-c(mylist, newlist) #concatenating the lists
expandedlist
```

```
$id
[1] 34453

$name
[1] "John"

$marks
[1] 13  3 12 15 19

$parents.names
[1] "Ana"  "Mike"

$age
[1] 19

$sex
[1] "male"
```

```
length(expandedlist)
```

```
[1] 6
```

Remove list components using negative index, or using NULL

Exercise:

Starting with the expanded list given above, what will be the result of the following statement? Consider the statement one by one.

```
expandedlist <- expandedlist[-5]
expandedlist #5th index is removed
```

```
$id
[1] 34453

$name
[1] "John"

$marks
[1] 13  3 12 15 19

$parents.names
[1] "Ana"  "Mike"

$sex
[1] "male"
```

```
expandedlist <- expandedlist[c(-1,-5)]
expandedlist #1st and 5th index elements removed
```

```
$name
[1] "John"

$marks
[1] 13  3 12 15 19
```

```
$parents.names
[1] "Ana"  "Mike"
```

```
expandedlist$parents.names <- NULL
expandedlist #parents name element is assigned to NULL. Hence its also removed
```

```
$name
[1] "John"

$marks
[1] 13  3 12 15 19
```

```
expandedlist[['marks']] <- NULL
expandedlist #similarly 'marks' are also removed
```

```
$name
[1] "John"
```

`unlist()` converts a list to a vector.

```
mylist
```

```
$id
[1] 34453

$name
[1] "John"

$marks
[1] 13  3 12 15 19

$parents.names
[1] "Ana"  "Mike"
```

```
unlist(mylist)
```

```
            id           name         marks1         marks2         marks3 
       "34453"         "John"           "13"            "3"           "12" 
        marks4         marks5 parents.names1 parents.names2 
          "15"           "19"          "Ana"         "Mike" 
```

```
mode(mylist)
```

```
[1] "list"
```

```
mode(unlist(mylist))
```

```
[1] "character"
```

```
is.vector(unlist(mylist)) #atomic list with names
```

```
[1] TRUE
```

```
is.list(mylist)
```

```
[1] TRUE
```

```
is.atomic(mylist)
```

```
[1] FALSE
```

```
is.list(unlist(mylist))
```

```
[1] FALSE
```

Data Frames

Data frames are a specific type of list: each row is an observation, and each column is an attribute. They are the recommended data format for tables (2-D).

The column names must not be empty, and the row names must be unique.

A data frame can store numeric, factor, or character data, and each column should have the same number of data elements.

Create a data frame

```
my.dataframe <- data.frame(site=c('A', 'B', 'A','A', 'B'), season=c('winter', 'summer', 's

my.dataframe
```

```
  site season  ph
1    A winter 7.4
2    B summer 6.3
3    A summer 8.6
4    A spring 7.2
5    B   fall 8.9
```

Different ways to access the elements in a dataframe (table): `[]`, `[[]]`, `$`,

Indexes and names

Exercise:

Given ‘my.dataframes‘, what values will the following statements access?

```
my.dataframe <- data.frame(site=c('A', 'B', 'A','A', 'B'), season=c('winter', 'summer', 's
```

```
my.dataframe[3, 2] #3rd row and 2nd column
```

```
[1] "summer"
```

```
my.dataframe[['site']] #print all the site elements
```

```
[1] "A" "B" "A" "A" "B"
```

```
my.dataframe['site'] #print the site elements in a df format
```

```
  site
1    A
2    B
3    A
4    A
5    B
```

```
my.dataframe[my.dataframe$ph>7, ] #print all entries whose ph>7
```

```
  site season  ph
1    A winter 7.4
3    A summer 8.6
4    A spring 7.2
5    B   fall 8.9
```

```
my.dataframe[my.dataframe$ph>7, 'site'] #print all sites whose ph>7
```

```
[1] "A" "A" "A" "B"
```

```
my.dataframe[my.dataframe$ph>7, c('site', 'ph')] #print all sites and it's ph whose ph>7
```

```
  site  ph
1    A 7.4
3    A 8.6
4    A 7.2
5    B 8.9
```

Use `subset()` to query a data frame

`subset()` can only query, it can not be used to change values in the data frame.

```
subset(my.dataframe, ph>7) #print all entries whose ph>7
```

```
  site season  ph
1    A winter 7.4
3    A summer 8.6
4    A spring 7.2
5    B   fall 8.9
```

```
subset(my.dataframe, ph>7, c("site", "ph"))
```

```
  site  ph
1    A 7.4
3    A 8.6
4    A 7.2
5    B 8.9
```

```
subset(my.dataframe[1:2,], ph>7, c(site, ph))
```

```
  site  ph
1    A 7.4
```

To change values in data frame - add 1 to `summer` `ph`:

```
#example
my.dataframe[my.dataframe$season=='summer', 'ph'] <- my.dataframe[my.dataframe$season=='su

my.dataframe[my.dataframe$season=='summer', 'ph'] #1 is added to ph values of summer
```

```
[1] 7.3 9.6
```

```
#example
my.dataframe[my.dataframe$season=='summer' & my.dataframe$ph>8, 'ph'] <- my.dataframe[my.d

my.dataframe[my.dataframe$season=='summer', 'ph']
```

```
[1]  7.3 10.6
```

Add a column

```
my.dataframe$NO3 <- c(234.5, 123.4, 456.7, 567.8, 789.0) #adding a new column

my.dataframe
```

```
  site season   ph   NO3
1    A winter  7.4 234.5
2    B summer  7.3 123.4
3    A summer 10.6 456.7
4    A spring  7.2 567.8
5    B   fall  8.9 789.0
```

Removing a column

```
#my.dataframe$NO3<-NULL
my.dataframe <- my.dataframe[, -4]
my.dataframe
```

```
  site season   ph
1    A winter  7.4
2    B summer  7.3
3    A summer 10.6
4    A spring  7.2
5    B   fall  8.9
```

Check the structure of a data frame:

```
str(my.dataframe)
```

```
'data.frame':	5 obs. of  3 variables:
 $ site  : chr  "A" "B" "A" "A" ...
 $ season: chr  "winter" "summer" "summer" "spring" ...
 $ ph    : num  7.4 7.3 10.6 7.2 8.9
```

```
nrow(my.dataframe) #no. of rows
```

```
[1] 5
```

```
ncol(my.dataframe) #no. of columns
```

```
[1] 3
```

```
dim(my.dataframe) #dimension
```

```
[1] 5 3
```

Edit a data frame:

```
 edit(my.dataframe) #this brings up a data editor

 View(my.dataframe) #this brings up a uneditable tab that display the data for you to view
```

Update names of the columns:

```
names(my.dataframe)
```

```
[1] "site"   "season" "ph"
```

```
names(my.dataframe) <- c('area', 'season', 'P.h.')
my.dataframe
```

```
  area season P.h.
1    A winter  7.4
2    B summer  7.3
3    A summer 10.6
4    A spring  7.2
5    B   fall  8.9
```

```
names(my.dataframe)[3] <- 'ph'
my.dataframe
```

```
  area season   ph
1    A winter  7.4
2    B summer  7.3
3    A summer 10.6
4    A spring  7.2
5    B   fall  8.9
```

Tibbles

Tibbles are like data frames, but they are more convenient.

Columns can be defined depending on already established columns. Tibbles cannot convert categorical valued attributes to factors and cannot print a whole data set .

```
install.packages("tibble")
                                                                  library(tibble)
```

Create a tibble

```
# Create a tibble called 'my.tibble' with three columns:
# 1. 'TempCels': A column of 100 random Celsius temperature values between -10 and 40.
# 2. 'TempFahr': A column that calculates Fahrenheit temperatures from 'TempCels' using th
# 3. 'Location': A column that repeats the letters 'a' and 'b' 50 times each.
my.tibble <- tibble(TempCels = sample(-10:40, size=100, replace=TRUE),
                    TempFahr = TempCels*9/5+32,
                    Location = rep(letters[1:2], each=50))

# Print the 'my.tibble' tibble to view the data.
my.tibble
```

```
# A tibble: 100 x 3
   TempCels TempFahr Location
      <int>    <dbl> <chr>
 1       -6     21.2 a
 2       34     93.2 a
 3        2     35.6 a
 4       25     77   a
 5       16     60.8 a
 6       19     66.2 a
 7       -2     28.4 a
 8        7     44.6 a
 9      -10     14   a
10        4     39.2 a
# i 90 more rows
```

Use the penguins data frame from the `palmerpenguins` package:

```
# Install the 'palmerpenguins' package if not already installed
install.packages("palmerpenguins")

# Load the 'palmerpenguins' package
library(palmerpenguins)

# Load the penguins dataset from the package
data(penguins)

# Check the dimensions of the dataset
dim(penguins)

# Check the class or data type of the dataset
class(penguins)

# Display the dataset
penguins
```

```
[1] 344   8
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```
# A tibble: 344 x 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
```

```
10 Adelie  Torgersen           42            20.2               190        4250
# i 334 more rows
# i 2 more variables: sex <fct>, year <int>
```

Convert a data frame to a tibble

```
# Convert the 'penguins' data frame to a tibble and store it in 'pe'
pe <- as_tibble(penguins)

# Check the class of the 'pe' object
class(pe)
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```
# Display pe
pe
```

```
# A tibble: 344 x 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# i 334 more rows
# i 2 more variables: sex <fct>, year <int>
```

A mode is a mutually exclusive classification of items based on their fundamental structure. Numeric, complex, character, and logical modes are the 'atomic' modes. Modes for recursive objects include 'list,' 'function,' and a few others. An item has exactly one mode.

A class is a property of an object that governs how generic functions interact with it. It is not a mutually exclusive category. By convention, if an object has no special class assigned to it, such as a simple numeric vector, its class is the same as its mode.

Changing the mode of an object is often called ‘coercion’. The mode of an object can change without necessarily changing the class.

e.g., typeof or specific type testers: is.vector, is.atomic, is.data.frame, etc.

```
x <- 1:16
mode(x)
```

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

```
dim(x) <- c(4,4)
class(x)
```

```
[1] "matrix" "array"
```

```
is.numeric(x)
```

```
[1] TRUE
```

```
mode(x) <- "character"
mode(x)
```

```
[1] "character"
```

```
class(x)
```

```
[1] "matrix" "array"
```

The mode changed from ‘numeric’ to ‘character’, but class stays ‘matrix’

However:

```
x <- factor(x)
class(x)
```

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

```r
mode(x)
```

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

class changed from ‘matrix’ to ‘factor’, but mode stays ‘numeric’ . At this stage, even though x has mode numeric again, its new class, ‘factor’, prohibits it being used in arithmetic operations.

A set of ‘is.xxx()‘ functions can be used to check the data structure of an object

```r
is.array(x)
```

```
[1] FALSE
```

```r
is.list(x)
```

```
[1] FALSE
```

```r
is.data.frame(x)
```

```
[1] FALSE
```

```r
is.matrix(x)
```

```
[1] FALSE
```

```r
is_tibble(x)
```

```
[1] FALSE
```

```r
is.vector(x)
```

```
[1] FALSE
```

```
typeof(x)
```

```
[1] "integer"
```

Subsetting a tibble results in a smaller tibble

```
# In the following lines, we're subsetting the data frames 'pe' and 'penguins'
# to select specific rows (1 to 15) and specific columns ("bill_length_mm" and "bill_depth

# In the first subset, we're selecting rows 1 to 15 and columns "bill_length_mm" and "bill
class(pe[1:15, c("bill_length_mm", "bill_depth_mm")])
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```
# In the second subset, we're doing the same for the 'penguins' data frame.
class(penguins[1:15, c("bill_length_mm", "bill_depth_mm")])
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```
# Now, in the next two lines, we're subsetting the same data frames 'pe' and 'penguins'
# but this time selecting only the "bill_length_mm" column.

# In the first subset, we're selecting rows 1 to 15 and only the "bill_length_mm" column f
class(pe[1:15, c("bill_length_mm")])
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```
# In the second subset, we're doing the same for the 'penguins' data frame.
class(penguins[1:15, c("bill_length_mm")])
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

dplyr

filter() vs. select()

`select()` selects a subset of columns of the dataset.

`filter()` select a subset of rows.

These two are often used in a nested fashion (like SQL SELECT/WHERE)

Other useful functions provided by `dplyr:mutate`, `summarise`, `arrange`, and joins (e.g,. `left_join()`, `right_join()`)

```r
install.packages("dplyr")
library(dplyr)
```

Select bill lengths and widths of species `Adelie`:

```r
select(filter(pe, species=="Adelie"), bill_length_mm, bill_depth_mm)
```

```
# A tibble: 152 x 2
   bill_length_mm bill_depth_mm
            <dbl>         <dbl>
 1           39.1          18.7
 2           39.5          17.4
 3           40.3          18
 4           NA            NA
 5           36.7          19.3
 6           39.3          20.6
 7           38.9          17.8
 8           39.2          19.6
 9           34.1          18.1
10           42            20.2
# i 142 more rows
```

```r
filter(select(pe, bill_length_mm, bill_depth_mm, species), species=="Adelie")
```

```
# A tibble: 152 x 3
   bill_length_mm bill_depth_mm species
            <dbl>         <dbl> <fct>
 1           39.1          18.7 Adelie
 2           39.5          17.4 Adelie
 3           40.3          18   Adelie
 4           NA            NA   Adelie
 5           36.7          19.3 Adelie
 6           39.3          20.6 Adelie
 7           38.9          17.8 Adelie
 8           39.2          19.6 Adelie
 9           34.1          18.1 Adelie
10           42            20.2 Adelie
# i 142 more rows
```

Exercise

How would you achieve the same result as the above but use tibble subsetting?

```
pe
```

```
# A tibble: 344 x 8
   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
 1 Adelie  Torgersen           39.1          18.7               181        3750
 2 Adelie  Torgersen           39.5          17.4               186        3800
 3 Adelie  Torgersen           40.3          18                 195        3250
 4 Adelie  Torgersen           NA            NA                  NA          NA
 5 Adelie  Torgersen           36.7          19.3               193        3450
 6 Adelie  Torgersen           39.3          20.6               190        3650
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# i 334 more rows
# i 2 more variables: sex <fct>, year <int>
```

```
#solution
# Method 1: Using bracket notation with subsetting
pe[pe$species == 'Adelie', c("bill_length_mm", "bill_depth_mm")]
```

```
# A tibble: 152 x 2
   bill_length_mm bill_depth_mm
            <dbl>         <dbl>
```

```
 1           39.1          18.7
 2           39.5          17.4
 3           40.3          18  
 4           NA            NA  
 5           36.7          19.3
 6           39.3          20.6
 7           38.9          17.8
 8           39.2          19.6
 9           34.1          18.1
10           42            20.2
# i 142 more rows
```

```
# Method 2: Using the 'subset' function
subset(pe, pe$species == 'Adelie', select = c("bill_length_mm", "bill_depth_mm"))
```

```
# A tibble: 152 x 2
   bill_length_mm bill_depth_mm
            <dbl>         <dbl>
 1           39.1          18.7
 2           39.5          17.4
 3           40.3          18  
 4           NA            NA  
 5           36.7          19.3
 6           39.3          20.6
 7           38.9          17.8
 8           39.2          19.6
 9           34.1          18.1
10           42            20.2
# i 142 more rows
```

Pipe |>, or the magrittr %>%, passes the output of a function to another function as its first argument.

```
select(pe, bill_length_mm, bill_depth_mm, species) |> filter(species=="Adelie")
```

```
# A tibble: 152 x 3
   bill_length_mm bill_depth_mm species
            <dbl>         <dbl> <fct>  
 1           39.1          18.7 Adelie 
 2           39.5          17.4 Adelie 
 3            40.3          18   Adelie
 4            NA            NA   Adelie
 5            36.7          19.3 Adelie
 6            39.3          20.6 Adelie
 7            38.9          17.8 Adelie
 8            39.2          19.6 Adelie
 9            34.1          18.1 Adelie
10            42            20.2 Adelie
# i 142 more rows
```

Exercise

Pass the result from the filter to the select function and achieve the same result as shown above.

```
filter(pe, species=="Adelie") |> select(bill_length_mm, bill_depth_mm, species)
```

```
# A tibble: 152 x 3
   bill_length_mm bill_depth_mm species
            <dbl>         <dbl> <fct>
 1           39.1          18.7 Adelie
 2           39.5          17.4 Adelie
 3           40.3          18   Adelie
 4           NA            NA   Adelie
 5           36.7          19.3 Adelie
 6           39.3          20.6 Adelie
 7           38.9          17.8 Adelie
 8           39.2          19.6 Adelie
 9           34.1          18.1 Adelie
10           42            20.2 Adelie
# i 142 more rows
```

Exercise

Create a data object to hold student names (Judy, Max, Dan) and their grades (78,85,99) Convert number grades to letter grades:90-100:A;80-89:B;70-79:C; \<70:F

```
# Create a list of students with names and grades
students <- list(names = c("Judy", "Max", "Dan"),
                 grades = c(78, 85, 99))

# Print the list before grade conversion
```

```r
print("Before:")
```

```
[1] "Before:"
```

```r
students
```

```
$names
[1] "Judy" "Max"  "Dan"

$grades
[1] 78 85 99
```

```r
# Define a function to convert numerical grades to letter grades
gradeConvertor <- function(grade) {
  # Convert the input grade to a numeric value (in case it's not already)
  grade <- as.numeric(grade)

  # Check if the grade is out of the valid range (0 to 100)
  if (grade > 100 | grade < 0)
    print("Grade out of the range")
  else if (grade >= 90 & grade <= 100)
    return("A")
  else if (grade >= 80 & grade < 90)
    return("B")
  else if (grade >= 70 & grade < 80)
    return("C")
  else
    return("F")
}

# Loop through the grades in the 'students' list and convert them
for (i in 1:length(students$grades)) {
  students$grades[i] <- gradeConvertor(students$grades[i])
}

# Print the list after grade conversion
print("After:")
```

```
[1] "After:"
```

```
students
```

```
$names
[1] "Judy" "Max"  "Dan"

$grades
[1] "C" "B" "A"
```