Basic programming

Stewart Li

8/29/2020

Document structure

1. R || Python || Javascript
2. Base R || Pandas
3. tidyverse and data.table

Variables

Variable types: character, numeric, integer, double, logical, date, factor, complex, raw, NULL.

```
class(c("strings"))
```

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

```
typeof(c(1L))
```

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

```
typeof(c(1))
```

```
## [1] "double"
```

```
typeof(c(TRUE))
```

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

```
class(as.Date(Sys.time()))
```

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

```
class(factor(c("hello", "hi", "bye", "bye")))
```

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

```
typeof(c(1i))
```

```
## [1] "complex"
```

```
class(c(charToRaw("strings")))
```

```
## [1] "raw"
```

`is` and `as`.

```
is.double(c(2))
```

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

```
is.nan(c(-2, 0, 2) / 0)
```

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

```
is.vector(numeric(10))
```

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

```
class(toString(c(2)))
```

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

```
class(as.numeric(c("2")))
```

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

Objects assignment: `=`, `<-`, `->`, `<<-`,

```
my.name <- readline(prompt="Enter name: ") # input in python
```

```
## Enter name:
```

```
print(paste("Hi,", my.name))
```

```
## [1] "Hi, "
```

Data structure

Vector: same data type, recycled, vectorised. Index and names: `$, [], [[]], .[]`.

```
c(11, "you", TRUE ) # without losing information
```

```
## [1] "11"   "you"  "TRUE"
```

```
c(11, TRUE )
```

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

```
c(c(1:3), c(1:10))
```

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

```
c(2, 4, 6)^2
```

```
## [1]  4 16 36
```

```
intersect(c(3:4), c(1:10))
```

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

```
union(c(11, 12), c(1:10))
```

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

```
numeric(10) # create a vector
```

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

```
character(10)
```

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

```
?`[` # i, j, k
```

```
## starting httpd help server ... done
```

```
a <- c(you = 3, me = 4)
names(a)
```

```
## [1] "you" "me"
```

```
tracemem(a) # id
```

```
## [1] "<0000000034259258>"
```

```
a['you'] # like counter in python
```

```
## you
##   3
```

```
a['them'] # NA
```

```
## <NA>
##   NA
```

```
a[1:2] # slice
```

```
## you  me
##   3   4
```

Matrices: same data type. Array: multiple dimensions.

```
matrix(letters, 13, 2, byrow = TRUE)
```

```
##       [,1] [,2]
##  [1,] "a"  "b"
##  [2,] "c"  "d"
##  [3,] "e"  "f"
##  [4,] "g"  "h"
##  [5,] "i"  "j"
##  [6,] "k"  "l"
##  [7,] "m"  "n"
##  [8,] "o"  "p"
##  [9,] "q"  "r"
## [10,] "s"  "t"
## [11,] "u"  "v"
## [12,] "w"  "x"
## [13,] "y"  "z"
```

```
solve(matrix(1:26, 2, 2)) # inverse
```

```
##      [,1] [,2]
## [1,]   -2  1.5
## [2,]    1 -0.5
```

```
matrix(1:26, 2, 2) %*% solve(matrix(1:26, 2, 2))
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

```
diag(2)
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    0    1
```

```
t(head(mtcars))
```

```
##      Mazda RX4 Mazda RX4 Wag Datsun 710 Hornet 4 Drive Hornet Sportabout
## mpg      21.00        21.000      22.80         21.400             18.70
## cyl       6.00         6.000       4.00          6.000              8.00
## disp    160.00       160.000     108.00        258.000            360.00
## hp      110.00       110.000      93.00        110.000            175.00
## drat      3.90         3.900       3.85          3.080              3.15
## wt        2.62         2.875       2.32          3.215              3.44
## qsec     16.46        17.020      18.61         19.440             17.02
## vs        0.00         0.000       1.00          1.000              0.00
## am        1.00         1.000       1.00          0.000              0.00
## gear      4.00         4.000       4.00          3.000              3.00
## carb      4.00         4.000       1.00          1.000              2.00
##      Valiant
## mpg    18.10
## cyl     6.00
## disp  225.00
## hp    105.00
## drat    2.76
## wt      3.46
## qsec   20.22
## vs      1.00
## am      0.00
## gear    3.00
## carb    1.00
```

List: anything. Dataframe: same length, rows and columns.

```
help("data.frame")

data.frame(id = 1:6,
           greet = sample(LETTERS, 6),
           nums = runif(6, 0, 100))
```

```
##   id greet      nums
## 1  1     F 90.788209
## 2  2     X 94.128634
## 3  3     Z 71.182116
## 4  4     M 90.497976
## 5  5     B  8.122285
## 6  6     V 76.072571
```

```
lista <- list(a = 1:10,
              b = c("you", "and", "me"),
              c = c(TRUE, FALSE),
              d = head(mtcars)) # like dict in python

lista[1] # extract
```

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

```
lista["a"]
```

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

```
lista[[1]]
```

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

```
lista$a
```

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

```
lista$e <- c(20:26) # create
append(lista, c(20:26)) # add
```

```
## $a
##  [1]  1  2  3  4  5  6  7  8  9 10
##
## $b
## [1] "you" "and" "me"
##
## $c
## [1]  TRUE FALSE
##
## $d
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
##
## $e
## [1] 20 21 22 23 24 25 26
##
## [[6]]
## [1] 20
##
## [[7]]
## [1] 21
##
## [[8]]
## [1] 22
##
## [[9]]
## [1] 23
##
## [[10]]
## [1] 24
##
## [[11]]
## [1] 25
##
## [[12]]
## [1] 26
```

```
lista[5] <- NULL # remove
lista[["a"]] <- c(13:19) # modify
```

Operators

Arithmetic calculator: `+, -, *, /, ^, %/%, %%` . Others: `;` , `#` ,

```
3 + 6
```

```
## [1] 9
```

```
`+` (3 + 6)
```

```
## [1] 9
```

```
49 %/% 2
```

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

Conditions

Logical operators: `==, !=, >, >=, |, &, !, %in%, -, xor` , Control flow: `if` , `switch` ,

```
"you" == "you"
```

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

```
"you" != "me"
```

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

```
if(49 > 49)
  {print('More')
}else if(49 > 50 | 0 >= -1){
    print("Less")
}else{
    print("None")
  }
```

```
## [1] "Less"
```

```
for (i in 1:10) {
  if(i %% 2 == 0){
    print(paste0("The number ", i, " is divisible by 2."))
  }else if(i %% 3 == 0){
    print(paste0("The number ", i, " is divisible by 3."))
  }else{
    print(paste0("The number ", i, " is not divisible by 2 or 3."))
  }
}
```

```
## [1] "The number 1 is not divisible by 2 or 3."
## [1] "The number 2 is divisible by 2."
## [1] "The number 3 is divisible by 3."
## [1] "The number 4 is divisible by 2."
## [1] "The number 5 is not divisible by 2 or 3."
## [1] "The number 6 is divisible by 2."
## [1] "The number 7 is not divisible by 2 or 3."
## [1] "The number 8 is divisible by 2."
## [1] "The number 9 is divisible by 3."
## [1] "The number 10 is divisible by 2."
```

Loop

```
for (i in 1:10) {
  print(i^2) # reset i each time
}
```

```
## [1] 1
## [1] 4
## [1] 9
## [1] 16
## [1] 25
## [1] 36
## [1] 49
## [1] 64
## [1] 81
## [1] 100
```

```
for (i in letters) {
  print(i)
}
```

```
## [1] "a"
## [1] "b"
## [1] "c"
## [1] "d"
## [1] "e"
## [1] "f"
## [1] "g"
## [1] "h"
## [1] "i"
## [1] "j"
## [1] "k"
## [1] "l"
## [1] "m"
## [1] "n"
## [1] "o"
## [1] "p"
## [1] "q"
## [1] "r"
## [1] "s"
## [1] "t"
## [1] "u"
## [1] "v"
## [1] "w"
## [1] "x"
## [1] "y"
## [1] "z"
```

```
for (i in seq_along(1:10)){
  print(paste0("this is ", i)) # index vs items), length(), seq_along()
}
```

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

```
ls() # last object
```

```
## [1] "a"       "i"       "lista"   "my.name"
```

Iterate over index or item.

```
a <- c(2, 5, "you")

for (x in a){
  if(x == "5"){
    print("you are so cool")
  }
}
```

```
## [1] "you are so cool"
```

```
for (x in 1:length(a)){
  if(a[x] == "5"){
    print("you are so cool")
  }
}
```

```
## [1] "you are so cool"
```

```
i <- 1:6
for (i in i) {
  if (i == 3){
    next
  }
  print(i)
}
```

```
## [1] 1
## [1] 2
## [1] 4
## [1] 5
## [1] 6
```

```
i <- 1
while (i < 6) {
  if (i == 3){
    break
  }
  print(i)
  i = i+1
}
```

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

Pre-allocation of object.

```
a <- vector(mode = 'list', length = 10)

for (i in 1:10) {
  a[i] <- i^2
}

unlist(a)
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```
a <- (1:10)
for (x in a){
  print(x^x)
}
```

```
## [1] 1
## [1] 4
## [1] 27
## [1] 256
## [1] 3125
## [1] 46656
## [1] 823543
## [1] 16777216
## [1] 387420489
## [1] 1e+10
```

```
c(1:10) %>% map_dbl(~.x^.x)
```

```
##  [1]           1           4          27         256        3125       46656
##  [7]      823543    16777216   387420489 10000000000
```

Functions

```
myfun <- function(x, y = 10){
  a <- x + y
  a <- paste0('result is a number of ', a)
  a
}

myfun(20)
```

```
## [1] "result is a number of 30"
```

```
myfun1 <- function(func, ...){
  func(mtcars$mpg)
} # input function

myfun1(mean)
```

```
## [1] 20.09062
```

```
myfun1(sum)
```

```
## [1] 642.9
```

```
myfun2 <- function(x){
  addmy <- function(){
    paste0("start ", mean(c(x, 10)), " end")
  }
  return(addmy)
} # output function

myfun2(90)()
```

```
## [1] "start 50 end"
```

Erorr handling

```
myupdate <- function(x){
    tryCatch(
      {
        y = x * 2
        return(y)
      },
      error = function(error_message) {
        message("This is my custom message.")
        cli::cli_alert_danger("what are you doing?")
        message("And below is the error message from R:")
        return(message(error_message)) # catch errors***
      }
    )
}

myupdate("a")
```

```
## This is my custom message.
```

```
## x what are you doing?
## And below is the error message from R:
```

OOP

```
Audit <- R6::R6Class(
  classname = "Audit",
  public = list(
    initialize = function(client, aic) {
      private$client <- client
      private$aic <- aic
      # if (!missing(client)) self$client <- client
      # if (!missing(aic)) self$aic <- aic
      self$greet()
    },
    set_aic = function(val) {
      private$aic <- val
    },
    greet = function() {
      cat(paste0(private$client, " is charge of the audit of ", private$aic, ".\n"))
    }
  ),
  private = list(
    client = NA,
    aic = NA
  )
)

nh2020 <- Audit$new("RAudit Solution LLP", "Stewart Li")
```

```
## RAudit Solution LLP is charge of the audit of Stewart Li.
```

```
nh2020$set_aic("Song Peng")
nh2020$greet()
```

```
## RAudit Solution LLP is charge of the audit of Song Peng.
```

Data sciense

1. File type: `.R` , `.Rmd` , `Rdata` ,
2. Import library: `dplyr::select` , `:::` ,

```
getwd()
```

```
## [1] "C:/Users/Stewart Li/Dropbox/0. Stewart publication_Always updated/stdatascience/
2020_summary/summary_tool/mytutorials"
```

```
dir(".")
```

```
##  [1] "0_introdution.Rmd"       "1_programming_r.html"
##  [3] "1_programming_r.knit.md" "1_programming_r.Rmd"
##  [5] "2_visulization.Rmd"      "3_manipulation.Rmd"
##  [7] "4_cleaning.Rmd"          "5_function.Rmd"
##  [9] "6_model.Rmd"             "7_report.Rmd"
## [11] "8_demo.Rmd"              "advancer.Rmd"
## [13] "assets"                  "data"
## [15] "for_loop.Rmd"            "hyflux"
## [17] "map.Rmd"                 "shiny.R"
## [19] "shinydashboard.R"
```

```
list.files()
```

```
##  [1] "0_introdution.Rmd"       "1_programming_r.html"
##  [3] "1_programming_r.knit.md" "1_programming_r.Rmd"
##  [5] "2_visulization.Rmd"      "3_manipulation.Rmd"
##  [7] "4_cleaning.Rmd"          "5_function.Rmd"
##  [9] "6_model.Rmd"             "7_report.Rmd"
## [11] "8_demo.Rmd"              "advancer.Rmd"
## [13] "assets"                  "data"
## [15] "for_loop.Rmd"            "hyflux"
## [17] "map.Rmd"                 "shiny.R"
## [19] "shinydashboard.R"
```

```
?mtcars

attributes(mtcars)
```

```
## $names
##  [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"   "gear"
## [11] "carb"
##
## $row.names
##  [1] "Mazda RX4"           "Mazda RX4 Wag"       "Datsun 710"
##  [4] "Hornet 4 Drive"      "Hornet Sportabout"   "Valiant"
##  [7] "Duster 360"          "Merc 240D"           "Merc 230"
## [10] "Merc 280"            "Merc 280C"           "Merc 450SE"
## [13] "Merc 450SL"          "Merc 450SLC"         "Cadillac Fleetwood"
## [16] "Lincoln Continental" "Chrysler Imperial"   "Fiat 128"
## [19] "Honda Civic"         "Toyota Corolla"      "Toyota Corona"
## [22] "Dodge Challenger"    "AMC Javelin"         "Camaro Z28"
## [25] "Pontiac Firebird"    "Fiat X1-9"           "Porsche 914-2"
## [28] "Lotus Europa"        "Ford Pantera L"      "Ferrari Dino"
## [31] "Maserati Bora"       "Volvo 142E"
##
## $class
## [1] "data.frame"
```

```
dim(mtcars)
```

```
## [1] 32 11
```

```
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

```
head(mtcars);tail(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
##                 mpg cyl  disp  hp drat    wt qsec vs am gear carb
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.7  0  1    5    2
## Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.9  1  1    5    2
## Ford Pantera L 15.8   8 351.0 264 4.22 3.170 14.5  0  1    5    4
## Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.5  0  1    5    6
## Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.6  0  1    5    8
## Volvo 142E     21.4   4 121.0 109 4.11 2.780 18.6  1  1    4    2
```

```
Map(min, mtcars)
```

```
## $mpg
## [1] 10.4
##
## $cyl
## [1] 4
##
## $disp
## [1] 71.1
##
## $hp
## [1] 52
##
## $drat
## [1] 2.76
##
## $wt
## [1] 1.513
##
## $qsec
## [1] 14.5
##
## $vs
## [1] 0
##
## $am
## [1] 0
##
## $gear
## [1] 3
##
## $carb
## [1] 1
```

```
apply(mtcars, 2, min)
```

```
##    mpg    cyl   disp     hp   drat     wt   qsec     vs     am   gear   carb
## 10.400  4.000 71.100 52.000  2.760  1.513 14.500  0.000  0.000  3.000  1.000
```

```
Reduce(`+`, mtcars)
```

```
##  [1] 328.980 329.795 259.580 426.135 590.310 385.540 656.920 270.980 299.570
## [10] 350.460 349.660 510.740 511.500 509.850 728.560 726.644 725.695 213.850
## [19] 195.165 206.955 273.775 519.650 506.085 646.280 631.175 208.215 272.570
## [28] 273.683 670.690 379.590 694.710 288.890
```

```
with(mtcars, by(am, gear, summary))
```

```
## gear: 3
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0       0       0       0       0
## ------------------------------------------------------------
## gear: 4
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  1.0000  0.6667  1.0000  1.0000
## ------------------------------------------------------------
## gear: 5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1       1       1       1       1       1
```

```
with(mtcars, aggregate(mpg, by = list(am, gear), FUN = function(x) c(mean(x), sd(x))))
```

```
##   Group.1 Group.2       x.1       x.2
## 1       0       3 16.106667  3.371618
## 2       0       4 21.050000  3.069745
## 3       1       4 26.275000  5.414465
## 4       1       5 21.380000  6.658979
```

```
mean(mtcars$mpg)
```

```
## [1] 20.09062
```

```
mean(mtcars[mtcars$hp > mean(mtcars$hp), "mpg"])
```

```
## [1] 15.40667
```

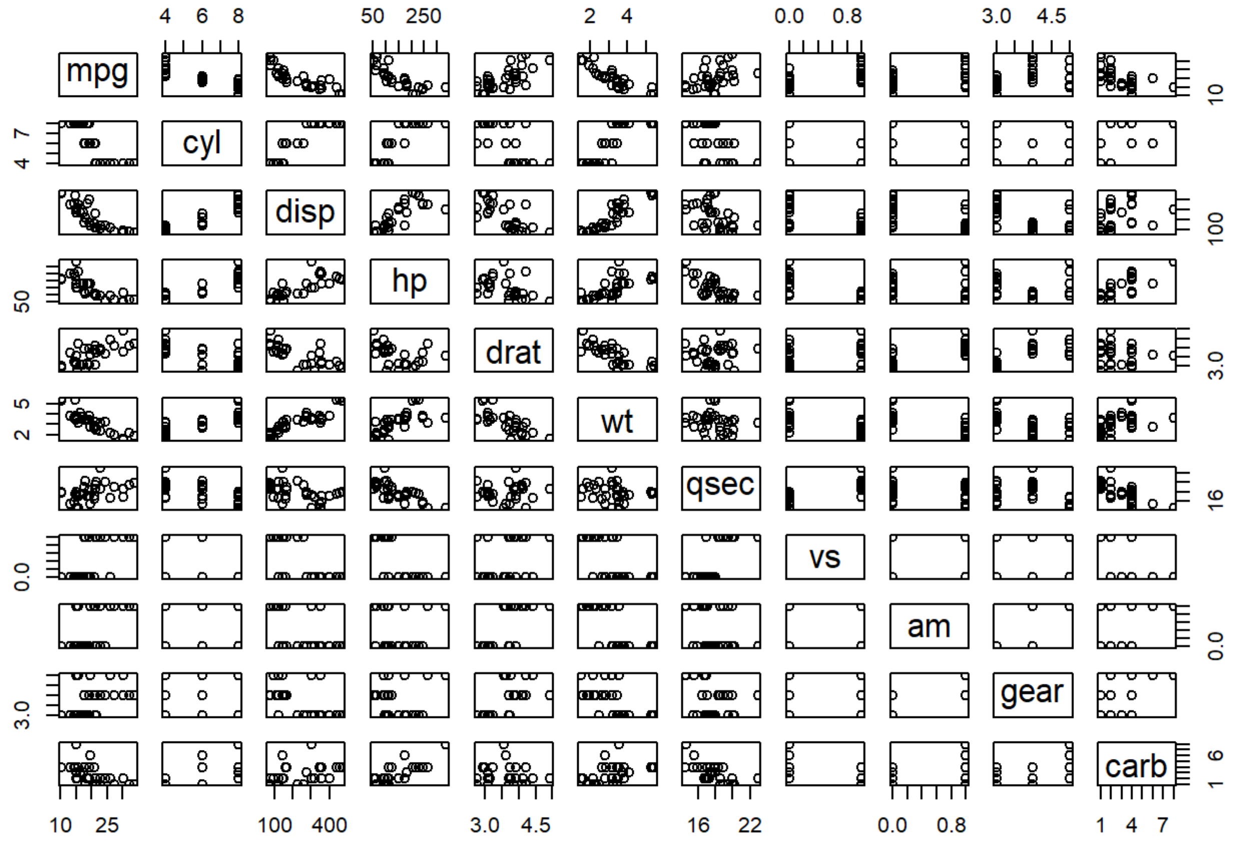
```
transform(mtcars, e = hp*.01)
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb    e
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4 1.10
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4 1.10
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1 0.93
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1 1.10
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2 1.75
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1 1.05
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4 2.45
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2 0.62
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2 0.95
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4 1.23
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4 1.23
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3 1.80
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3 1.80
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3 1.80
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4 2.05
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4 2.15
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4 2.30
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1 0.66
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2 0.52
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1 0.65
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1 0.97
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2 1.50
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2 1.50
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4 2.45
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2 1.75
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1 0.66
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2 0.91
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2 1.13
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4 2.64
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6 1.75
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8 3.35
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2 1.09
```

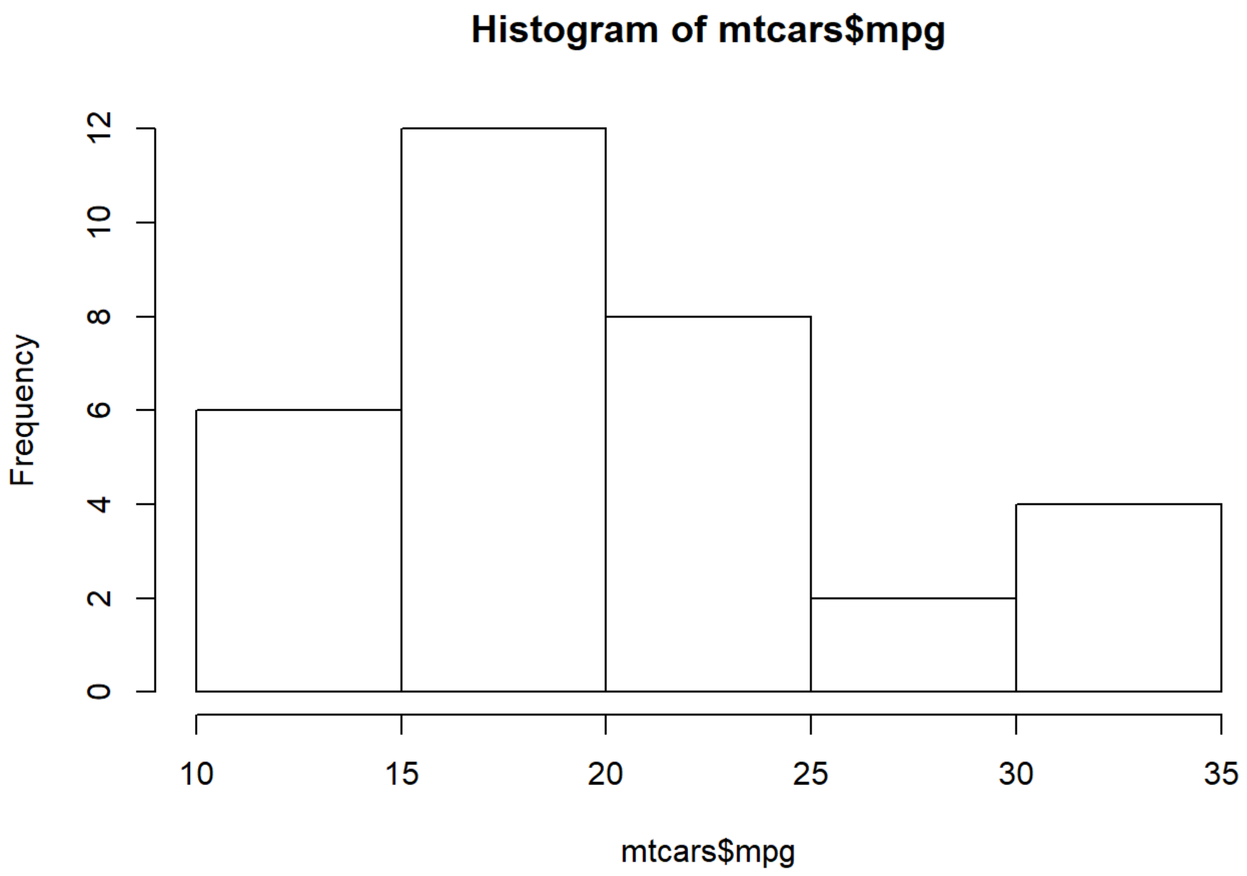
```
merge(mtcars, data.frame(am = c("yes", "no")), by = "am", all = TRUE) # row bind
```

```
##      am  mpg cyl  disp  hp drat    wt  qsec vs gear carb
## 1     0 21.4   6 258.0 110 3.08 3.215 19.44  1    3    1
## 2     0 18.7   8 360.0 175 3.15 3.440 17.02  0    3    2
## 3     0 18.1   6 225.0 105 2.76 3.460 20.22  1    3    1
## 4     0 14.3   8 360.0 245 3.21 3.570 15.84  0    3    4
## 5     0 24.4   4 146.7  62 3.69 3.190 20.00  1    4    2
## 6     0 22.8   4 140.8  95 3.92 3.150 22.90  1    4    2
## 7     0 19.2   6 167.6 123 3.92 3.440 18.30  1    4    4
## 8     0 17.8   6 167.6 123 3.92 3.440 18.90  1    4    4
## 9     0 16.4   8 275.8 180 3.07 4.070 17.40  0    3    3
## 10    0 17.3   8 275.8 180 3.07 3.730 17.60  0    3    3
## 11    0 15.2   8 275.8 180 3.07 3.780 18.00  0    3    3
## 12    0 10.4   8 472.0 205 2.93 5.250 17.98  0    3    4
## 13    0 10.4   8 460.0 215 3.00 5.424 17.82  0    3    4
## 14    0 14.7   8 440.0 230 3.23 5.345 17.42  0    3    4
## 15    0 21.5   4 120.1  97 3.70 2.465 20.01  1    3    1
## 16    0 15.5   8 318.0 150 2.76 3.520 16.87  0    3    2
## 17    0 15.2   8 304.0 150 3.15 3.435 17.30  0    3    2
## 18    0 13.3   8 350.0 245 3.73 3.840 15.41  0    3    4
## 19    0 19.2   8 400.0 175 3.08 3.845 17.05  0    3    2
## 20    1 21.0   6 160.0 110 3.90 2.620 16.46  0    4    4
## 21    1 21.0   6 160.0 110 3.90 2.875 17.02  0    4    4
## 22    1 22.8   4 108.0  93 3.85 2.320 18.61  1    4    1
## 23    1 32.4   4  78.7  66 4.08 2.200 19.47  1    4    1
## 24    1 30.4   4  75.7  52 4.93 1.615 18.52  1    4    2
## 25    1 33.9   4  71.1  65 4.22 1.835 19.90  1    4    1
## 26    1 27.3   4  79.0  66 4.08 1.935 18.90  1    4    1
## 27    1 26.0   4 120.3  91 4.43 2.140 16.70  0    5    2
## 28    1 30.4   4  95.1 113 3.77 1.513 16.90  1    5    2
## 29    1 15.8   8 351.0 264 4.22 3.170 14.50  0    5    4
## 30    1 19.7   6 145.0 175 3.62 2.770 15.50  0    5    6
## 31    1 15.0   8 301.0 335 3.54 3.570 14.60  0    5    8
## 32    1 21.4   4 121.0 109 4.11 2.780 18.60  1    4    2
## 33  no   NA  NA    NA  NA   NA    NA    NA NA   NA   NA
## 34 yes   NA  NA    NA  NA   NA    NA    NA NA   NA   NA
```

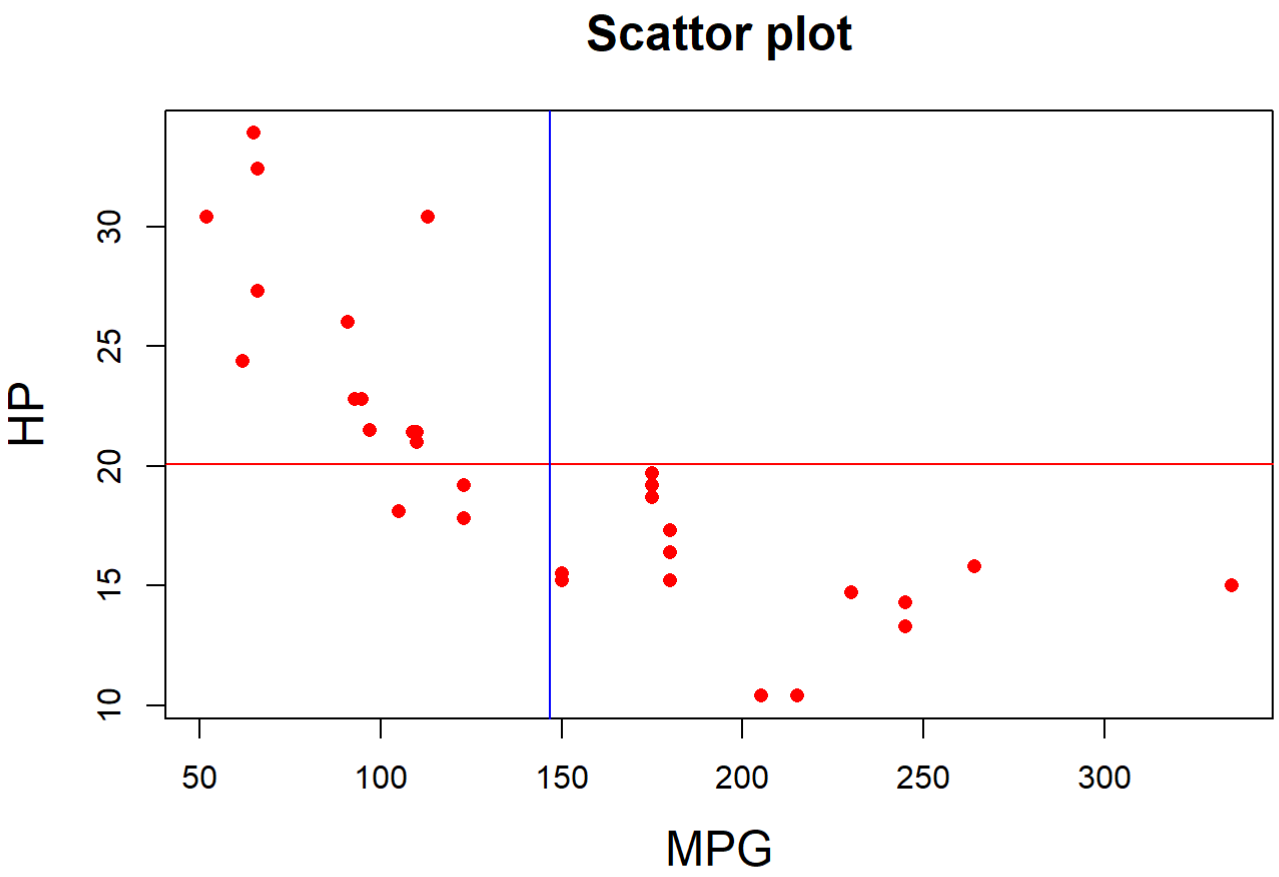
```
plot(mtcars) # pairs()
```



```
hist(mtcars$mpg)
```



```
plot(mpg ~ hp,
     data = mtcars,
     main = 'Scattor plot',
     xlab = 'MPG',
     ylab = 'HP',
     cex.main = 1.5,
     cex.lab = 1.5,
     col = 'red',
     pch = 16)
abline(h = mean(mtcars$mpg), col = 'red')
abline(v = mean(mtcars$hp), col = 'blue')
```


Scattor plot
HP
10
15
20
25
30
MPG
50
100
150
200
250
300