Day3.R

RKC

2024-10-27

```
#### USer Defined Functions ####

# First User Def Function
first_fn = function(){
  print("Print First Function")
}

first_fn()
```

```
## [1] "Print First Function"
```

```
# Adding Function
add = function(x,y){
  print(x+y)
}

add(2,3)
```

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

```
# User input using function
Add = function(){
  x = as.numeric(readline("Enter First NUmber ---> "))
  y = as.numeric(readline("Enter second NUmber ---> "))
  print(x+y)
}

Add()
```

```
## Enter First NUmber --->
## Enter second NUmber --->
## [1] NA
```

```
#### Control Statements ####

# If condition
x = 2
if (x>0) {
  print(paste(x, " is positive number"))
}
```

```
## [1] "2  is positive number"
```

```
# If else condition
x = 2
if (x<0) {
  print(paste(x, ' is negative number'))
} else{
  cat(x, ' is positive number') # Don't user cat with print
}
```

```
## 2  is positive number
```

```
# if else if condition

x = '+ab'
class(x)
```

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

```
if (x>0) {
  print(paste(x, 'is positive number'))
} else if(x<0){
  print(paste(x, 'is negative number'))
}else{
  print(paste(x, 'is neither positive nor negative'))
}
```

```
## [1] "+ab is negative number"
```

```
x = 21
if (x %% 2 == 0) {
  print("NUmber is even")
}else{
  print(paste(x, 'Number is Odd'))
}
```

```
## [1] "21 Number is Odd"
```

```
### LOOPS ###

# For Loop

x = letters[4:10]
for (i in x) {
  print(i)
}
```

```
## [1] "d"
## [1] "e"
## [1] "f"
## [1] "g"
## [1] "h"
## [1] "i"
## [1] "j"
```

```r
# print table for 7:22
x = c(1:22)
a = 0
for (i in x) {
  print(i * 7)
}
```

```
## [1] 7
## [1] 14
## [1] 21
## [1] 28
## [1] 35
## [1] 42
## [1] 49
## [1] 56
## [1] 63
## [1] 70
## [1] 77
## [1] 84
## [1] 91
## [1] 98
## [1] 105
## [1] 112
## [1] 119
## [1] 126
## [1] 133
## [1] 140
## [1] 147
## [1] 154
```

```r
# Print1, 1.5,2, 2.5

for (i in c(1:20)) {
  i = i / 2
  print(i+0.5)
}
```

```
## [1] 1
## [1] 1.5
## [1] 2
## [1] 2.5
## [1] 3
## [1] 3.5
## [1] 4
## [1] 4.5
## [1] 5
## [1] 5.5
## [1] 6
## [1] 6.5
## [1] 7
## [1] 7.5
## [1] 8
## [1] 8.5
```

```
## [1] 9
## [1] 9.5
## [1] 10
## [1] 10.5
```

```
# While Loop

x = 2
while (x<=5) {
  print(x)
  x = x+1
}
```

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

```
# Print multiplication table of 7 upto 7*22 using while loop

num = 1
while (num<= 22) {
  print(paste("7 x", num, "=", 7*num))
  num = num +1
}
```

```
## [1] "7 x 1 = 7"
## [1] "7 x 2 = 14"
## [1] "7 x 3 = 21"
## [1] "7 x 4 = 28"
## [1] "7 x 5 = 35"
## [1] "7 x 6 = 42"
## [1] "7 x 7 = 49"
## [1] "7 x 8 = 56"
## [1] "7 x 9 = 63"
## [1] "7 x 10 = 70"
## [1] "7 x 11 = 77"
## [1] "7 x 12 = 84"
## [1] "7 x 13 = 91"
## [1] "7 x 14 = 98"
## [1] "7 x 15 = 105"
## [1] "7 x 16 = 112"
## [1] "7 x 17 = 119"
## [1] "7 x 18 = 126"
## [1] "7 x 19 = 133"
## [1] "7 x 20 = 140"
## [1] "7 x 21 = 147"
## [1] "7 x 22 = 154"
```

```
# sum of first n natural numbers
n = 20
sum = 0
x = 0
```

```
while(x<n){
  x = x+1
  sum = sum + x
  print(sum)
}
```

```
## [1] 1
## [1] 3
## [1] 6
## [1] 10
## [1] 15
## [1] 21
## [1] 28
## [1] 36
## [1] 45
## [1] 55
## [1] 66
## [1] 78
## [1] 91
## [1] 105
## [1] 120
## [1] 136
## [1] 153
## [1] 171
## [1] 190
## [1] 210
```

```
# Repeat Statement print 1:7

x = 1
repeat{
  print(x)
  x = x+1
  if (x>7){
    break
  }
}
```

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

```
# print mul of 7:22

x = 1
repeat{
  print((7*x))
  x = x + 1
```

```
  if(x>=22){
    break
  }
}
```

```
## [1] 7
## [1] 14
## [1] 21
## [1] 28
## [1] 35
## [1] 42
## [1] 49
## [1] 56
## [1] 63
## [1] 70
## [1] 77
## [1] 84
## [1] 91
## [1] 98
## [1] 105
## [1] 112
## [1] 119
## [1] 126
## [1] 133
## [1] 140
## [1] 147
```

```
# NEXT LOOP

x = 1:17
for (i in x){
  if (i%%2 != 0){
    next
  }
  print(i)
}
```

```
## [1] 2
## [1] 4
## [1] 6
## [1] 8
## [1] 10
## [1] 12
## [1] 14
## [1] 16
```

```
#print odd numbers using next statement

x = 1:20
for (i in x){
  if(i%%2 == 0){
    next
```

```
  }
  print(i)
}
```

```
## [1] 1
## [1] 3
## [1] 5
## [1] 7
## [1] 9
## [1] 11
## [1] 13
## [1] 15
## [1] 17
## [1] 19
```

```
# SWITCH CASE

#?switch
#  case = action

operation = 'mul'
result = switch(operation , add = 5+3, sub=5-3, mul = 5*3); result
```

```
## [1] 15
```

```
# Function calculator from users to create calulator

calculator = function(){
  operation = readline("What operation would you like to perform --> ")
  x = as.numeric(readline("Enter first numer --> "))
  y = as.numeric(readline("Enter second numbver ---> "))
  result = switch(operation, add=x+y, sub= x-y, mul = x*y, div = x%/%y )
  print(result)
}
calculator()
```

```
## What operation would you like to perform -->
## Enter first numer -->
## Enter second numbver --->
## NULL
```

```
# Factorial
#
# factorial = function(){
#   input = as.numeric(readline("Enter a Whole NUmber to find its Factorial-->"))
#   if (input > 0) {
#     input1 = c(1:input)
#     n = 1
#     for (i in input1) {
#       n = n * i
#     }
#     print(n)
```

```
#   }
#   else if(input < 0 ){
#     print("Input is Invalid, Enter Whole number")
#     }else if (input == 0){
#     print('1')
#     }
# }
# factorial()


#write a user define function to calculate simple interest and compound interest

simpleinterest = function(){
  p = as.numeric(readline("Enter Initial Principal Balance --> "))
  r = as.double(readline("Enter Annual rate of interest --> "))
  t = as.numeric(readline("Enter Time period (in years) --> "))

  si = (p * ((r*t)))/ 100
  print(si)
}

simpleinterest()
```

```
## Enter Initial Principal Balance -->
## Enter Annual rate of interest -->
## Enter Time period (in years) -->
## [1] NA
```

```
#
# compoundinterest = function(){
#   p = as.numeric(readline("Enter Initial Principal Balance --> "))
#   r = as.double(readline("Enter Annual rate of interest --> "))
#   t = as.numeric(readline("Enter Time period (in years) --> "))
#
#   n = c(1:t)
#   ci = 0
#   for (i in n) {
#     ci =  (p * ((1+(r/100)**(n*t)) - p ))
#   }
#      #(p * (1+(r/n))**(n*t)) / 100
#   #}
#   print(ci)
# }

# Compounded Annually Formula   A = P (1 + r)t
#
# compoundinterest()
#
# compoundinterest <- function() {
#   # Getting user inputs
#   p <- as.numeric(readline("Enter Initial Principal Balance --> "))
#   r <- as.double(readline("Enter Annual Rate of Interest (%) --> "))
#   t <- as.numeric(readline("Enter Time Period (in years) --> "))
```

```r
#
#   # Initializing compound interest variable
#   ci <- p * ((1 + (r / 100))^t - 1)
#
#   # Printing the compound interest
#   print(paste("Compound Interest after", t, "years is:", ci))
# }

# Run the function
# compoundinterest()

#write a user define fn to calculate simple and compound interest
#
# interest=function(){
#   p=as.numeric(readline("Enter Amount: "))
#   rate=as.double(readline("Enter Rate of interest: "))
#   t=as.numeric(readline("Enter number of years: "))
#
#   simple=(p*rate*t)/100
#   print(paste("Simple interest is ",simple))
#
#   amt=(p*(1+(rate/100))**t)
#   cmpd=amt-p
#   print(paste("Compound interest is ",cmpd))
# }
#
# interest()

#Write user define function in R to take temperature from user in
#degree Celsius and convert in kelvin and farenheit #Write a function
# "count elements"that accept a numeric vector and counts the number of
# positive, negative and zero elements

#test the function on vector

# define a function to take four numbers from user and print thenumber in ascending
# order without using sort
#
# sorting = function(){
#   print("Enter four numbers :: ")
#   n = scan(nmax = 4)
#   print(n)
#
# }
# n = scan(nmax = 4)
# print(n)
# print(class(n))

#### TIDYVERSE ####

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
```

```
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ----------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
View(diamonds)

?diamonds
```

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

```
# PIPE OPERATOR
library(magrittr)
```

```
##
## Attaching package: 'magrittr'
##
## The following object is masked _by_ '.GlobalEnv':
##
##     add
##
## The following object is masked from 'package:purrr':
##
##     set_names
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
iris %>% head()
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
diamonds %>% head()
```

```
## # A tibble: 6 x 10
##   carat cut       color clarity depth table price     x     y     z
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
## 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
```

```
## 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
## 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
## 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
## 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
```

```
## BASIC DATA MANAGEMENT CHAPTER - pg 65

# MUTATE()
# What it does: Adds new columns or modifies current variables in the dataset.

class(diamonds)  # [1] "tbl_df"     "tbl"        "data.frame"
```

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

```
# Added 3 new columns - until its assigned  it won't affect base data
diamonds %>%
        mutate(Justone = 1,
               Values = 'something',
               Simple= TRUE) %>% head()
```

```
## # A tibble: 6 x 13
##   carat cut     color clarity depth table price     x     y     z Justone Values
##   <dbl> <ord>   <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>   <dbl> <chr>
## 1  0.23 Ideal   E     SI2      61.5    55   326  3.95  3.98  2.43       1 somet~
## 2  0.21 Premium E     SI1      59.8    61   326  3.89  3.84  2.31       1 somet~
## 3  0.23 Good    E     VS1      56.9    65   327  4.05  4.07  2.31       1 somet~
## 4  0.29 Premium I     VS2      62.4    58   334  4.2   4.23  2.63       1 somet~
## 5  0.31 Good    J     SI2      63.3    58   335  4.34  4.35  2.75       1 somet~
## 6  0.24 Very G~ J     VVS2     62.8    57   336  3.94  3.96  2.48       1 somet~
## # i 1 more variable: Simple <lgl>
```

```
# adding new calculative columns
diamonds %>%
  mutate(price200 = price - 200,
        price20perc = price * 0.20,
        price20percoff = price * 0.80,
        pricepercarat = price / carat,
        pizza = depth ^ 2) %>% head()
```

```
## # A tibble: 6 x 15
##   carat cut       color clarity depth table price     x     y     z price200
##   <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>    <dbl>
## 1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43      126
## 2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31      126
## 3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31      127
## 4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63      134
## 5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75      135
## 6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48      136
## # i 4 more variables: price20perc <dbl>, price20percoff <dbl>,
## #   pricepercarat <dbl>, pizza <dbl>
```

```r
# SUMMARIZE - collapses all rows and returns a one-row summary. R will recognize both the British and
# American spelling (summarise/summarize).
# mean won't make sense to add it as column hence we
# use summarize just as "describe()" in python

# getting mean
diamonds %>% summarize(avg.price = mean(price))
```

```
## # A tibble: 1 x 1
##   avg.price
##       <dbl>
## 1     3933.
```

```r
# getting stdev
diamonds %>% summarize(avg.price = sd(price))
```

```
## # A tibble: 1 x 1
##   avg.price
##       <dbl>
## 1     3989.
```

```r
# getting median
diamonds %>% summarize(avg.price = median(price))
```

```
## # A tibble: 1 x 1
##   avg.price
##       <dbl>
## 1      2401
```

```r
# GROUP BY FUNCTION - group_by() : Takes existing data and groups specific
# variables together for future operations. Many operations are performed
# on groups.

diamonds %>%
          group_by(cut) %>%
                        #Grouped by price on cut column
                        summarize(m = mean(price), n = n()) %>% ungroup()
```

```
## # A tibble: 5 x 3
##   cut           m     n
##   <ord>     <dbl> <int>
## 1 Fair      4359.  1610
## 2 Good      3929.  4906
## 3 Very Good 3982. 12082
## 4 Premium   4584. 13791
## 5 Ideal     3458. 21551
```

```r
                                                # for safer side ungroup
### Creating table from page 73 from book
```

```r
## Creating identification number to represent 50 individual people
ID <- c(1:50)
## Creating sex variable (25 males/25 females)
Sex <- rep(c("male", "female"), 25) # rep stands for replicate
## Creating age variable (20-39 year olds)
Age <- c(26, 25, 39, 37, 31, 34, 34, 30, 26, 33,
         39, 28, 26, 29, 33, 22, 35, 23, 26, 36,
         21, 20, 31, 21, 35, 39, 36, 22, 22, 25,
         27, 30, 26, 34, 38, 39, 30, 29, 26, 25,
         26, 36, 23, 21, 21, 39, 26, 26, 27, 21)
## Creating a dependent variable called Score
Score <- c(0.010, 0.418, 0.014, 0.090, 0.061, 0.328, 0.656, 0.002, 0.639, 0.173,
           0.076, 0.152, 0.467, 0.186, 0.520, 0.493, 0.388, 0.501, 0.800, 0.482,
           0.384, 0.046, 0.920, 0.865, 0.625, 0.035, 0.501, 0.851, 0.285, 0.752,
           0.686, 0.339, 0.710, 0.665, 0.214, 0.560, 0.287, 0.665, 0.630, 0.567,
           0.812, 0.637, 0.772, 0.905, 0.405, 0.363, 0.773, 0.410, 0.535, 0.449)
## Creating a unified dataset that puts together all variables
data <- tibble(ID, Sex, Age, Score)
View(data)

# Grouping only by Sex
data %>%
  group_by(Sex) %>%
  summarize(m = mean(Score), # calculates the mean
            s = sd(Score), # calculates the standard deviation
            n = n()) %>% # calculates the total number of observations
  ungroup()
```

```
## # A tibble: 2 x 4
##   Sex        m     s     n
##   <chr>  <dbl> <dbl> <int>
## 1 female 0.437 0.268    25
## 2 male   0.487 0.268    25
```

```r
# Grouping by Sex and Age
data %>%
  group_by(Sex, Age) %>% # grouped by Sex and Age
  summarize(m = mean(Score),
            s = sd(Score),
            n = n()) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'Sex'. You can override using the `.groups`
## argument.
```

```
## # A tibble: 27 x 5
##    Sex      Age     m      s     n
##    <chr>  <dbl> <dbl>  <dbl> <int>
##  1 female    20 0.046 NA         1
##  2 female    21 0.740  0.253     3
##  3 female    22 0.672  0.253     2
##  4 female    23 0.501 NA         1
```

```
##  5 female    25 0.579  0.167     3
##  6 female    26 0.41   NA        1
##  7 female    28 0.152  NA        1
##  8 female    29 0.426  0.339     2
##  9 female    30 0.170  0.238     2
## 10 female    33 0.173  NA        1
## # i 17 more rows
```

```
# FILTER() - Only retain specific rows of data that meet the specified requirement(s).

diamonds %>% filter(cut == 'Fair')
```

```
## # A tibble: 1,610 x 10
##    carat cut   color clarity depth table price     x     y     z
##    <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.22 Fair  E     VS2      65.1    61   337  3.87  3.78  2.49
##  2  0.86 Fair  E     SI2      55.1    69  2757  6.45  6.33  3.52
##  3  0.96 Fair  F     SI2      66.3    62  2759  6.27  5.95  4.07
##  4  0.7  Fair  F     VS2      64.5    57  2762  5.57  5.53  3.58
##  5  0.7  Fair  F     VS2      65.3    55  2762  5.63  5.58  3.66
##  6  0.91 Fair  H     SI2      64.4    57  2763  6.11  6.09  3.93
##  7  0.91 Fair  H     SI2      65.7    60  2763  6.03  5.99  3.95
##  8  0.98 Fair  H     SI2      67.9    60  2777  6.05  5.97  4.08
##  9  0.84 Fair  G     SI1      55.1    67  2782  6.39  6.2   3.47
## 10  1.01 Fair  E     I1       64.5    58  2788  6.29  6.21  4.03
## # i 1,600 more rows
```

```
data %>% filter(Sex == 'female' )
```

```
## # A tibble: 25 x 4
##       ID Sex      Age Score
##    <int> <chr>  <dbl> <dbl>
##  1     2 female    25 0.418
##  2     4 female    37 0.09
##  3     6 female    34 0.328
##  4     8 female    30 0.002
##  5    10 female    33 0.173
##  6    12 female    28 0.152
##  7    14 female    29 0.186
##  8    16 female    22 0.493
##  9    18 female    23 0.501
## 10    20 female    36 0.482
## # i 15 more rows
```

```
diamonds %>% filter(cut == 'Fair' | cut == 'Good', price <= 600)
```

```
## # A tibble: 505 x 10
##    carat cut   color clarity depth table price     x     y     z
##    <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Good  E     VS1      56.9    65   327  4.05  4.07  2.31
##  2  0.31 Good  J     SI2      63.3    58   335  4.34  4.35  2.75
##  3  0.22 Fair  E     VS2      65.1    61   337  3.87  3.78  2.49
```

```
##  4  0.3  Good  J     SI1      64      55   339  4.25  4.28  2.73
##  5  0.3  Good  J     SI1      63.4    54   351  4.23  4.29  2.7
##  6  0.3  Good  J     SI1      63.8    56   351  4.23  4.26  2.71
##  7  0.3  Good  I     SI2      63.3    56   351  4.26  4.3   2.71
##  8  0.23 Good  F     VS1      58.2    59   402  4.06  4.08  2.37
##  9  0.23 Good  E     VS1      64.1    59   402  3.83  3.85  2.46
## 10  0.31 Good  H     SI1      64      54   402  4.29  4.31  2.75
## # i 495 more rows
```

```
# Filtering
diamonds %>% filter(cut %in% c('Fair', 'Good'), price <= 600)
```

```
## # A tibble: 505 x 10
##    carat cut   color clarity depth table price     x     y     z
##    <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Good  E     VS1      56.9    65   327  4.05  4.07  2.31
##  2  0.31 Good  J     SI2      63.3    58   335  4.34  4.35  2.75
##  3  0.22 Fair  E     VS2      65.1    61   337  3.87  3.78  2.49
##  4  0.3  Good  J     SI1      64      55   339  4.25  4.28  2.73
##  5  0.3  Good  J     SI1      63.4    54   351  4.23  4.29  2.7
##  6  0.3  Good  J     SI1      63.8    56   351  4.23  4.26  2.71
##  7  0.3  Good  I     SI2      63.3    56   351  4.26  4.3   2.71
##  8  0.23 Good  F     VS1      58.2    59   402  4.06  4.08  2.37
##  9  0.23 Good  E     VS1      64.1    59   402  3.83  3.85  2.46
## 10  0.31 Good  H     SI1      64      54   402  4.29  4.31  2.75
## # i 495 more rows
```

```
# won't give any o/p as and (,) wont work
diamonds %>% filter(cut == 'Fair', cut == 'Good', price <= 600)
```

```
## # A tibble: 0 x 10
## # i 10 variables: carat <dbl>, cut <ord>, color <ord>, clarity <ord>,
## #   depth <dbl>, table <dbl>, price <int>, x <dbl>, y <dbl>, z <dbl>
```

```
diamonds %>% filter(cut == 'Fair' & cut == 'Good', price <= 600)
```

```
## # A tibble: 0 x 10
## # i 10 variables: carat <dbl>, cut <ord>, color <ord>, clarity <ord>,
## #   depth <dbl>, table <dbl>, price <int>, x <dbl>, y <dbl>, z <dbl>
```

```
diamonds %>% filter(cut == 'Fair' , depth < 60,  price <= 600)
```

```
## # A tibble: 3 x 10
##   carat cut   color clarity depth table price     x     y     z
##   <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.31 Fair  E     SI1      56.9    66   579  4.53  4.47  2.56
## 2  0.31 Fair  E     VS2      55.9    62   581  4.6   4.56  2.56
## 3  0.25 Fair  E     VS1      55.2    64   361  4.21  4.23  2.33
```

```
diamonds %>% filter(price %in% seq(c(300:650)))
```

```
## # A tibble: 21 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
## # i 11 more rows
```

```
View(diamonds)

# Incerase cut price by 200+ where cut == fair
diamonds %>% mutate(price = price +200) %>% filter(cut == 'Fair')
```

```
## # A tibble: 1,610 x 10
##    carat cut   color clarity depth table price     x     y     z
##    <dbl> <ord> <ord> <ord>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  0.22 Fair  E     VS2      65.1    61   537  3.87  3.78  2.49
##  2  0.86 Fair  E     SI2      55.1    69  2957  6.45  6.33  3.52
##  3  0.96 Fair  F     SI2      66.3    62  2959  6.27  5.95  4.07
##  4  0.7  Fair  F     VS2      64.5    57  2962  5.57  5.53  3.58
##  5  0.7  Fair  F     VS2      65.3    55  2962  5.63  5.58  3.66
##  6  0.91 Fair  H     SI2      64.4    57  2963  6.11  6.09  3.93
##  7  0.91 Fair  H     SI2      65.7    60  2963  6.03  5.99  3.95
##  8  0.98 Fair  H     SI2      67.9    60  2977  6.05  5.97  4.08
##  9  0.84 Fair  G     SI1      55.1    67  2982  6.39  6.2   3.47
## 10  1.01 Fair  E     I1       64.5    58  2988  6.29  6.21  4.03
## # i 1,600 more rows
```

```
#Check below conditon
# diamonds %>% filter(cut == 'Fair') %>%  mutate(price = price +200)

### SELECT() FUNCTION just like SQL
# Select only the columns (variables) that you want to see. Gets rid of all other columns. You can
# to refer to the columns by the column position (first column) or by name. The order in which you list
# column names/positions is the order that the columns will be displayed.

# column names / position is the order that the column
diamonds %>% select(cut, color)
```

```
## # A tibble: 53,940 x 2
##    cut       color
##    <ord>     <ord>
##  1 Ideal     E
```

```
##  2 Premium   E
##  3 Good      E
##  4 Premium   I
##  5 Good      J
##  6 Very Good J
##  7 Very Good I
##  8 Very Good H
##  9 Fair      E
## 10 Very Good H
## # i 53,930 more rows
```

```
# filters column
diamonds %>% select(1:5)
```

```
## # A tibble: 53,940 x 5
##    carat cut       color clarity depth
##    <dbl> <ord>     <ord> <ord>   <dbl>
##  1  0.23 Ideal     E     SI2      61.5
##  2  0.21 Premium   E     SI1      59.8
##  3  0.23 Good      E     VS1      56.9
##  4  0.29 Premium   I     VS2      62.4
##  5  0.31 Good      J     SI2      63.3
##  6  0.24 Very Good J     VVS2     62.8
##  7  0.24 Very Good I     VVS1     62.3
##  8  0.26 Very Good H     SI1      61.9
##  9  0.22 Fair      E     VS2      65.1
## 10  0.23 Very Good H     VS1      59.4
## # i 53,930 more rows
```

```
# columms filters
diamonds %>% select(1,2,3)
```

```
## # A tibble: 53,940 x 3
##    carat cut       color
##    <dbl> <ord>     <ord>
##  1  0.23 Ideal     E
##  2  0.21 Premium   E
##  3  0.23 Good      E
##  4  0.29 Premium   I
##  5  0.31 Good      J
##  6  0.24 Very Good J
##  7  0.24 Very Good I
##  8  0.26 Very Good H
##  9  0.22 Fair      E
## 10  0.23 Very Good H
## # i 53,930 more rows
```

```
# Displays all cols except 'cut'
diamonds %>% select(-cut)
```

```
## # A tibble: 53,940 x 9
##    carat color clarity depth table price     x     y     z
```

```
##    <dbl> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 H     VS1      59.4    61   338  4     4.05  2.39
## # i 53,930 more rows
```

```
# Displays all cols except 'cut' and 'color
diamonds %>% select(-cut, -color)
```

```
## # A tibble: 53,940 x 8
##    carat clarity depth table price     x     y     z
##    <dbl> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 VS1      59.4    61   338  4     4.05  2.39
## # i 53,930 more rows
```

```
# OR
diamonds %>% select(-(1:5))
```

```
## # A tibble: 53,940 x 5
##    table price     x     y     z
##    <dbl> <int> <dbl> <dbl> <dbl>
##  1    55   326  3.95  3.98  2.43
##  2    61   326  3.89  3.84  2.31
##  3    65   327  4.05  4.07  2.31
##  4    58   334  4.2   4.23  2.63
##  5    58   335  4.34  4.35  2.75
##  6    57   336  3.94  3.96  2.48
##  7    57   336  3.95  3.98  2.47
##  8    55   337  4.07  4.11  2.53
##  9    61   337  3.87  3.78  2.49
## 10    61   338  4     4.05  2.39
## # i 53,930 more rows
```

```
### ARRANGE FUNCTION
#
# Allows you arrange values within a variable in ascending or descending order
# (if that is applicable to your values). This can apply to both numerical
```

```
# and non-numerical values.

# Ascending order
diamonds %>% arrange(cut)
```

```
## # A tibble: 53,940 x 10
##    carat cut   color clarity depth table price     x     y     z
##    <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.22 Fair  E     VS2      65.1    61   337  3.87  3.78  2.49
##  2  0.86 Fair  E     SI2      55.1    69  2757  6.45  6.33  3.52
##  3  0.96 Fair  F     SI2      66.3    62  2759  6.27  5.95  4.07
##  4  0.7  Fair  F     VS2      64.5    57  2762  5.57  5.53  3.58
##  5  0.7  Fair  F     VS2      65.3    55  2762  5.63  5.58  3.66
##  6  0.91 Fair  H     SI2      64.4    57  2763  6.11  6.09  3.93
##  7  0.91 Fair  H     SI2      65.7    60  2763  6.03  5.99  3.95
##  8  0.98 Fair  H     SI2      67.9    60  2777  6.05  5.97  4.08
##  9  0.84 Fair  G     SI1      55.1    67  2782  6.39  6.2   3.47
## 10  1.01 Fair  E     I1       64.5    58  2788  6.29  6.21  4.03
## # i 53,930 more rows
```

```
diamonds %>% arrange(price)
```

```
## # A tibble: 53,940 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
## # i 53,930 more rows
```

```
# Descending
diamonds %>% arrange(desc(cut))
```

```
## # A tibble: 53,940 x 10
##    carat cut   color clarity depth table price     x     y     z
##    <dbl> <ord> <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.23 Ideal J     VS1      62.8    56   340  3.93  3.9   2.46
##  3  0.31 Ideal J     SI2      62.2    54   344  4.35  4.37  2.71
##  4  0.3  Ideal I     SI2      62      54   348  4.31  4.34  2.68
##  5  0.33 Ideal I     SI2      61.8    55   403  4.49  4.51  2.78
##  6  0.33 Ideal I     SI2      61.2    56   403  4.49  4.5   2.75
##  7  0.33 Ideal J     SI1      61.1    56   403  4.49  4.55  2.76
##  8  0.23 Ideal G     VS1      61.9    54   404  3.93  3.95  2.44
##  9  0.32 Ideal I     SI1      60.9    55   404  4.45  4.48  2.72
```

```
## 10  0.3  Ideal I     SI2      61      59   405  4.3   4.33  2.63
## # i 53,930 more rows
```

```
diamonds %>% arrange(desc(price))
```

```
## # A tibble: 53,940 x 10
##    carat cut       color clarity depth table price     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  2.29 Premium   I     VS2      60.8    60 18823  8.5   8.47  5.16
##  2  2    Very Good G     SI1      63.5    56 18818  7.9   7.97  5.04
##  3  1.51 Ideal     G     IF       61.7    55 18806  7.37  7.41  4.56
##  4  2.07 Ideal     G     SI2      62.5    55 18804  8.2   8.13  5.11
##  5  2    Very Good H     SI1      62.8    57 18803  7.95  8     5.01
##  6  2.29 Premium   I     SI1      61.8    59 18797  8.52  8.45  5.24
##  7  2.04 Premium   H     SI1      58.1    60 18795  8.37  8.28  4.84
##  8  2    Premium   I     VS1      60.8    59 18795  8.13  8.02  4.91
##  9  1.71 Premium   F     VS2      62.3    59 18791  7.57  7.53  4.7
## 10  2.15 Ideal     G     SI2      62.6    54 18791  8.29  8.35  5.21
## # i 53,930 more rows
```

```
### count() function - Collapses the rows and counts the number of observations per group of values.

diamonds %>% count(cut)
```

```
## # A tibble: 5 x 2
##   cut           n
##   <ord>     <int>
## 1 Fair       1610
## 2 Good       4906
## 3 Very Good 12082
## 4 Premium   13791
## 5 Ideal     21551
```

```
diamonds %>% group_by(cut) %>% count()
```

```
## # A tibble: 5 x 2
## # Groups:   cut [5]
##   cut           n
##   <ord>     <int>
## 1 Fair       1610
## 2 Good       4906
## 3 Very Good 12082
## 4 Premium   13791
## 5 Ideal     21551
```

```
### RENAME() Function

# can rename column name
diamonds %>% rename(PRICE = price)
```

```
## # A tibble: 53,940 x 10
```

```
##    carat cut       color clarity depth table PRICE     x     y     z
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326  3.95  3.98  2.43
##  2  0.21 Premium   E     SI1      59.8    61   326  3.89  3.84  2.31
##  3  0.23 Good      E     VS1      56.9    65   327  4.05  4.07  2.31
##  4  0.29 Premium   I     VS2      62.4    58   334  4.2   4.23  2.63
##  5  0.31 Good      J     SI2      63.3    58   335  4.34  4.35  2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336  3.94  3.96  2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336  3.95  3.98  2.47
##  8  0.26 Very Good H     SI1      61.9    55   337  4.07  4.11  2.53
##  9  0.22 Fair      E     VS2      65.1    61   337  3.87  3.78  2.49
## 10  0.23 Very Good H     VS1      59.4    61   338  4     4.05  2.39
## # i 53,930 more rows
```

```
diamonds %>% rename(length = x, width = y, depths = z)
```

```
## # A tibble: 53,940 x 10
##    carat cut       color clarity depth table price length width depths
##    <dbl> <ord>     <ord> <ord>   <dbl> <dbl> <int>  <dbl> <dbl>  <dbl>
##  1  0.23 Ideal     E     SI2      61.5    55   326   3.95  3.98   2.43
##  2  0.21 Premium   E     SI1      59.8    61   326   3.89  3.84   2.31
##  3  0.23 Good      E     VS1      56.9    65   327   4.05  4.07   2.31
##  4  0.29 Premium   I     VS2      62.4    58   334   4.2   4.23   2.63
##  5  0.31 Good      J     SI2      63.3    58   335   4.34  4.35   2.75
##  6  0.24 Very Good J     VVS2     62.8    57   336   3.94  3.96   2.48
##  7  0.24 Very Good I     VVS1     62.3    57   336   3.95  3.98   2.47
##  8  0.26 Very Good H     SI1      61.9    55   337   4.07  4.11   2.53
##  9  0.22 Fair      E     VS2      65.1    61   337   3.87  3.78   2.49
## 10  0.23 Very Good H     VS1      59.4    61   338   4     4.05   2.39
## # i 53,930 more rows
```

```
### RECODE() function - pg 71

Sex = factor(c('male', 'm', 'M', 'Female', 'female', 'Female'))
TestScore = c(10,20,10,25,12,5)

dataset = tibble(Sex, TestScore)
str(dataset)
```

```
## tibble [6 x 2] (S3: tbl_df/tbl/data.frame)
##  $ Sex      : Factor w/ 5 levels "female","Female",..: 5 3 4 2 1 2
##  $ TestScore: num [1:6] 10 20 10 25 12 5
```

```
# Creating a new variable (Sex.new) with recoded values
# from the original variable (Sex)


dataset %>%
        mutate(
          Sex.new = recode(Sex, "m" = "Male", "M"="Male", 'male' = 'Male')
        )
```

```
## # A tibble: 6 x 3
##   Sex    TestScore Sex.new
##   <fct>      <dbl> <fct>
## 1 male          10 Male
## 2 m             20 Male
## 3 M             10 Male
## 4 Female        25 Female
## 5 female        12 female
## 6 Female         5 Female
```

```
### ROW_NUMBER()

# Using row_number() with mutate() will create a column of consecutive numbers
# the row_number() fucntion is useful for creating an identification number
# (an ID variable), it is also useful for labeling each


# Load the inbuilt iris dataset and display  first 15 rows
df = iris
  # Whats is the structure of the dataset?
  str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Filter the dataset to show only the species "setosa"
  df %>% filter(Species == 'setosa')
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1           5.1         3.5          1.4         0.2  setosa
## 2           4.9         3.0          1.4         0.2  setosa
## 3           4.7         3.2          1.3         0.2  setosa
## 4           4.6         3.1          1.5         0.2  setosa
## 5           5.0         3.6          1.4         0.2  setosa
## 6           5.4         3.9          1.7         0.4  setosa
## 7           4.6         3.4          1.4         0.3  setosa
## 8           5.0         3.4          1.5         0.2  setosa
## 9           4.4         2.9          1.4         0.2  setosa
## 10          4.9         3.1          1.5         0.1  setosa
## 11          5.4         3.7          1.5         0.2  setosa
## 12          4.8         3.4          1.6         0.2  setosa
## 13          4.8         3.0          1.4         0.1  setosa
## 14          4.3         3.0          1.1         0.1  setosa
## 15          5.8         4.0          1.2         0.2  setosa
## 16          5.7         4.4          1.5         0.4  setosa
## 17          5.4         3.9          1.3         0.4  setosa
## 18          5.1         3.5          1.4         0.3  setosa
## 19          5.7         3.8          1.7         0.3  setosa
## 20          5.1         3.8          1.5         0.3  setosa
```

```
## 21          5.4         3.4          1.7         0.2  setosa
## 22          5.1         3.7          1.5         0.4  setosa
## 23          4.6         3.6          1.0         0.2  setosa
## 24          5.1         3.3          1.7         0.5  setosa
## 25          4.8         3.4          1.9         0.2  setosa
## 26          5.0         3.0          1.6         0.2  setosa
## 27          5.0         3.4          1.6         0.4  setosa
## 28          5.2         3.5          1.5         0.2  setosa
## 29          5.2         3.4          1.4         0.2  setosa
## 30          4.7         3.2          1.6         0.2  setosa
## 31          4.8         3.1          1.6         0.2  setosa
## 32          5.4         3.4          1.5         0.4  setosa
## 33          5.2         4.1          1.5         0.1  setosa
## 34          5.5         4.2          1.4         0.2  setosa
## 35          4.9         3.1          1.5         0.2  setosa
## 36          5.0         3.2          1.2         0.2  setosa
## 37          5.5         3.5          1.3         0.2  setosa
## 38          4.9         3.6          1.4         0.1  setosa
## 39          4.4         3.0          1.3         0.2  setosa
## 40          5.1         3.4          1.5         0.2  setosa
## 41          5.0         3.5          1.3         0.3  setosa
## 42          4.5         2.3          1.3         0.3  setosa
## 43          4.4         3.2          1.3         0.2  setosa
## 44          5.0         3.5          1.6         0.6  setosa
## 45          5.1         3.8          1.9         0.4  setosa
## 46          4.8         3.0          1.4         0.3  setosa
## 47          5.1         3.8          1.6         0.2  setosa
## 48          4.6         3.2          1.4         0.2  setosa
## 49          5.3         3.7          1.5         0.2  setosa
## 50          5.0         3.3          1.4         0.2  setosa
```

```
  # Hw many rows aer there for species "setosa"
  df %>% filter(Species == 'setosa') %>% count()
```

```
##    n
## 1 50
```

```
  # Display the first 10 rows of the filtered dataset
  df %>% filter(Species == 'setosa') %>% head(10)
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1           5.1         3.5          1.4         0.2  setosa
## 2           4.9         3.0          1.4         0.2  setosa
## 3           4.7         3.2          1.3         0.2  setosa
## 4           4.6         3.1          1.5         0.2  setosa
## 5           5.0         3.6          1.4         0.2  setosa
## 6           5.4         3.9          1.7         0.4  setosa
## 7           4.6         3.4          1.4         0.3  setosa
## 8           5.0         3.4          1.5         0.2  setosa
## 9           4.4         2.9          1.4         0.2  setosa
## 10          4.9         3.1          1.5         0.1  setosa
```

```r
# select Sepal.length and species column
  df[c(1,5)]
```

```
##     Sepal.Length    Species
## 1            5.1     setosa
## 2            4.9     setosa
## 3            4.7     setosa
## 4            4.6     setosa
## 5            5.0     setosa
## 6            5.4     setosa
## 7            4.6     setosa
## 8            5.0     setosa
## 9            4.4     setosa
## 10           4.9     setosa
## 11           5.4     setosa
## 12           4.8     setosa
## 13           4.8     setosa
## 14           4.3     setosa
## 15           5.8     setosa
## 16           5.7     setosa
## 17           5.4     setosa
## 18           5.1     setosa
## 19           5.7     setosa
## 20           5.1     setosa
## 21           5.4     setosa
## 22           5.1     setosa
## 23           4.6     setosa
## 24           5.1     setosa
## 25           4.8     setosa
## 26           5.0     setosa
## 27           5.0     setosa
## 28           5.2     setosa
## 29           5.2     setosa
## 30           4.7     setosa
## 31           4.8     setosa
## 32           5.4     setosa
## 33           5.2     setosa
## 34           5.5     setosa
## 35           4.9     setosa
## 36           5.0     setosa
## 37           5.5     setosa
## 38           4.9     setosa
## 39           4.4     setosa
## 40           5.1     setosa
## 41           5.0     setosa
## 42           4.5     setosa
## 43           4.4     setosa
## 44           5.0     setosa
## 45           5.1     setosa
## 46           4.8     setosa
## 47           5.1     setosa
## 48           4.6     setosa
## 49           5.3     setosa
```

```
## 50          5.0     setosa
## 51          7.0 versicolor
## 52          6.4 versicolor
## 53          6.9 versicolor
## 54          5.5 versicolor
## 55          6.5 versicolor
## 56          5.7 versicolor
## 57          6.3 versicolor
## 58          4.9 versicolor
## 59          6.6 versicolor
## 60          5.2 versicolor
## 61          5.0 versicolor
## 62          5.9 versicolor
## 63          6.0 versicolor
## 64          6.1 versicolor
## 65          5.6 versicolor
## 66          6.7 versicolor
## 67          5.6 versicolor
## 68          5.8 versicolor
## 69          6.2 versicolor
## 70          5.6 versicolor
## 71          5.9 versicolor
## 72          6.1 versicolor
## 73          6.3 versicolor
## 74          6.1 versicolor
## 75          6.4 versicolor
## 76          6.6 versicolor
## 77          6.8 versicolor
## 78          6.7 versicolor
## 79          6.0 versicolor
## 80          5.7 versicolor
## 81          5.5 versicolor
## 82          5.5 versicolor
## 83          5.8 versicolor
## 84          6.0 versicolor
## 85          5.4 versicolor
## 86          6.0 versicolor
## 87          6.7 versicolor
## 88          6.3 versicolor
## 89          5.6 versicolor
## 90          5.5 versicolor
## 91          5.5 versicolor
## 92          6.1 versicolor
## 93          5.8 versicolor
## 94          5.0 versicolor
## 95          5.6 versicolor
## 96          5.7 versicolor
## 97          5.7 versicolor
## 98          6.2 versicolor
## 99          5.1 versicolor
## 100         5.7 versicolor
## 101         6.3  virginica
## 102         5.8  virginica
## 103         7.1  virginica
```

```
## 104          6.3  virginica
## 105          6.5  virginica
## 106          7.6  virginica
## 107          4.9  virginica
## 108          7.3  virginica
## 109          6.7  virginica
## 110          7.2  virginica
## 111          6.5  virginica
## 112          6.4  virginica
## 113          6.8  virginica
## 114          5.7  virginica
## 115          5.8  virginica
## 116          6.4  virginica
## 117          6.5  virginica
## 118          7.7  virginica
## 119          7.7  virginica
## 120          6.0  virginica
## 121          6.9  virginica
## 122          5.6  virginica
## 123          7.7  virginica
## 124          6.3  virginica
## 125          6.7  virginica
## 126          7.2  virginica
## 127          6.2  virginica
## 128          6.1  virginica
## 129          6.4  virginica
## 130          7.2  virginica
## 131          7.4  virginica
## 132          7.9  virginica
## 133          6.4  virginica
## 134          6.3  virginica
## 135          6.1  virginica
## 136          7.7  virginica
## 137          6.3  virginica
## 138          6.4  virginica
## 139          6.0  virginica
## 140          6.9  virginica
## 141          6.7  virginica
## 142          6.9  virginica
## 143          5.8  virginica
## 144          6.8  virginica
## 145          6.7  virginica
## 146          6.7  virginica
## 147          6.3  virginica
## 148          6.5  virginica
## 149          6.2  virginica
## 150          5.9  virginica
```

```
  #Create a new dataframe called setosa_sepal with these column
  setosa_sepal = tibble(df[c(1,5)]); setosa_sepal
```

```
## # A tibble: 150 x 2
##    Sepal.Length Species
##           <dbl> <fct>
```

```
##  1          5.1 setosa
##  2          4.9 setosa
##  3          4.7 setosa
##  4          4.6 setosa
##  5          5   setosa
##  6          5.4 setosa
##  7          4.6 setosa
##  8          5   setosa
##  9          4.4 setosa
## 10          4.9 setosa
## # i 140 more rows
```

```
# Create a new column Sepal.Area in the Iris Dataset that calculates the area
# of Sepal(Area = Width * length)
  df %>% mutate(
    Sepal.Area = (Sepal.Width * Sepal.Length)
  )
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species Sepal.Area
## 1            5.1         3.5          1.4         0.2     setosa      17.85
## 2            4.9         3.0          1.4         0.2     setosa      14.70
## 3            4.7         3.2          1.3         0.2     setosa      15.04
## 4            4.6         3.1          1.5         0.2     setosa      14.26
## 5            5.0         3.6          1.4         0.2     setosa      18.00
## 6            5.4         3.9          1.7         0.4     setosa      21.06
## 7            4.6         3.4          1.4         0.3     setosa      15.64
## 8            5.0         3.4          1.5         0.2     setosa      17.00
## 9            4.4         2.9          1.4         0.2     setosa      12.76
## 10           4.9         3.1          1.5         0.1     setosa      15.19
## 11           5.4         3.7          1.5         0.2     setosa      19.98
## 12           4.8         3.4          1.6         0.2     setosa      16.32
## 13           4.8         3.0          1.4         0.1     setosa      14.40
## 14           4.3         3.0          1.1         0.1     setosa      12.90
## 15           5.8         4.0          1.2         0.2     setosa      23.20
## 16           5.7         4.4          1.5         0.4     setosa      25.08
## 17           5.4         3.9          1.3         0.4     setosa      21.06
## 18           5.1         3.5          1.4         0.3     setosa      17.85
## 19           5.7         3.8          1.7         0.3     setosa      21.66
## 20           5.1         3.8          1.5         0.3     setosa      19.38
## 21           5.4         3.4          1.7         0.2     setosa      18.36
## 22           5.1         3.7          1.5         0.4     setosa      18.87
## 23           4.6         3.6          1.0         0.2     setosa      16.56
## 24           5.1         3.3          1.7         0.5     setosa      16.83
## 25           4.8         3.4          1.9         0.2     setosa      16.32
## 26           5.0         3.0          1.6         0.2     setosa      15.00
## 27           5.0         3.4          1.6         0.4     setosa      17.00
## 28           5.2         3.5          1.5         0.2     setosa      18.20
## 29           5.2         3.4          1.4         0.2     setosa      17.68
## 30           4.7         3.2          1.6         0.2     setosa      15.04
## 31           4.8         3.1          1.6         0.2     setosa      14.88
## 32           5.4         3.4          1.5         0.4     setosa      18.36
## 33           5.2         4.1          1.5         0.1     setosa      21.32
## 34           5.5         4.2          1.4         0.2     setosa      23.10
## 35           4.9         3.1          1.5         0.2     setosa      15.19
```

```
## 36          5.0         3.2          1.2         0.2     setosa      16.00
## 37          5.5         3.5          1.3         0.2     setosa      19.25
## 38          4.9         3.6          1.4         0.1     setosa      17.64
## 39          4.4         3.0          1.3         0.2     setosa      13.20
## 40          5.1         3.4          1.5         0.2     setosa      17.34
## 41          5.0         3.5          1.3         0.3     setosa      17.50
## 42          4.5         2.3          1.3         0.3     setosa      10.35
## 43          4.4         3.2          1.3         0.2     setosa      14.08
## 44          5.0         3.5          1.6         0.6     setosa      17.50
## 45          5.1         3.8          1.9         0.4     setosa      19.38
## 46          4.8         3.0          1.4         0.3     setosa      14.40
## 47          5.1         3.8          1.6         0.2     setosa      19.38
## 48          4.6         3.2          1.4         0.2     setosa      14.72
## 49          5.3         3.7          1.5         0.2     setosa      19.61
## 50          5.0         3.3          1.4         0.2     setosa      16.50
## 51          7.0         3.2          4.7         1.4 versicolor      22.40
## 52          6.4         3.2          4.5         1.5 versicolor      20.48
## 53          6.9         3.1          4.9         1.5 versicolor      21.39
## 54          5.5         2.3          4.0         1.3 versicolor      12.65
## 55          6.5         2.8          4.6         1.5 versicolor      18.20
## 56          5.7         2.8          4.5         1.3 versicolor      15.96
## 57          6.3         3.3          4.7         1.6 versicolor      20.79
## 58          4.9         2.4          3.3         1.0 versicolor      11.76
## 59          6.6         2.9          4.6         1.3 versicolor      19.14
## 60          5.2         2.7          3.9         1.4 versicolor      14.04
## 61          5.0         2.0          3.5         1.0 versicolor      10.00
## 62          5.9         3.0          4.2         1.5 versicolor      17.70
## 63          6.0         2.2          4.0         1.0 versicolor      13.20
## 64          6.1         2.9          4.7         1.4 versicolor      17.69
## 65          5.6         2.9          3.6         1.3 versicolor      16.24
## 66          6.7         3.1          4.4         1.4 versicolor      20.77
## 67          5.6         3.0          4.5         1.5 versicolor      16.80
## 68          5.8         2.7          4.1         1.0 versicolor      15.66
## 69          6.2         2.2          4.5         1.5 versicolor      13.64
## 70          5.6         2.5          3.9         1.1 versicolor      14.00
## 71          5.9         3.2          4.8         1.8 versicolor      18.88
## 72          6.1         2.8          4.0         1.3 versicolor      17.08
## 73          6.3         2.5          4.9         1.5 versicolor      15.75
## 74          6.1         2.8          4.7         1.2 versicolor      17.08
## 75          6.4         2.9          4.3         1.3 versicolor      18.56
## 76          6.6         3.0          4.4         1.4 versicolor      19.80
## 77          6.8         2.8          4.8         1.4 versicolor      19.04
## 78          6.7         3.0          5.0         1.7 versicolor      20.10
## 79          6.0         2.9          4.5         1.5 versicolor      17.40
## 80          5.7         2.6          3.5         1.0 versicolor      14.82
## 81          5.5         2.4          3.8         1.1 versicolor      13.20
## 82          5.5         2.4          3.7         1.0 versicolor      13.20
## 83          5.8         2.7          3.9         1.2 versicolor      15.66
## 84          6.0         2.7          5.1         1.6 versicolor      16.20
## 85          5.4         3.0          4.5         1.5 versicolor      16.20
## 86          6.0         3.4          4.5         1.6 versicolor      20.40
## 87          6.7         3.1          4.7         1.5 versicolor      20.77
## 88          6.3         2.3          4.4         1.3 versicolor      14.49
## 89          5.6         3.0          4.1         1.3 versicolor      16.80
```

```
## 90          5.5         2.5          4.0         1.3 versicolor      13.75
## 91          5.5         2.6          4.4         1.2 versicolor      14.30
## 92          6.1         3.0          4.6         1.4 versicolor      18.30
## 93          5.8         2.6          4.0         1.2 versicolor      15.08
## 94          5.0         2.3          3.3         1.0 versicolor      11.50
## 95          5.6         2.7          4.2         1.3 versicolor      15.12
## 96          5.7         3.0          4.2         1.2 versicolor      17.10
## 97          5.7         2.9          4.2         1.3 versicolor      16.53
## 98          6.2         2.9          4.3         1.3 versicolor      17.98
## 99          5.1         2.5          3.0         1.1 versicolor      12.75
## 100         5.7         2.8          4.1         1.3 versicolor      15.96
## 101         6.3         3.3          6.0         2.5  virginica      20.79
## 102         5.8         2.7          5.1         1.9  virginica      15.66
## 103         7.1         3.0          5.9         2.1  virginica      21.30
## 104         6.3         2.9          5.6         1.8  virginica      18.27
## 105         6.5         3.0          5.8         2.2  virginica      19.50
## 106         7.6         3.0          6.6         2.1  virginica      22.80
## 107         4.9         2.5          4.5         1.7  virginica      12.25
## 108         7.3         2.9          6.3         1.8  virginica      21.17
## 109         6.7         2.5          5.8         1.8  virginica      16.75
## 110         7.2         3.6          6.1         2.5  virginica      25.92
## 111         6.5         3.2          5.1         2.0  virginica      20.80
## 112         6.4         2.7          5.3         1.9  virginica      17.28
## 113         6.8         3.0          5.5         2.1  virginica      20.40
## 114         5.7         2.5          5.0         2.0  virginica      14.25
## 115         5.8         2.8          5.1         2.4  virginica      16.24
## 116         6.4         3.2          5.3         2.3  virginica      20.48
## 117         6.5         3.0          5.5         1.8  virginica      19.50
## 118         7.7         3.8          6.7         2.2  virginica      29.26
## 119         7.7         2.6          6.9         2.3  virginica      20.02
## 120         6.0         2.2          5.0         1.5  virginica      13.20
## 121         6.9         3.2          5.7         2.3  virginica      22.08
## 122         5.6         2.8          4.9         2.0  virginica      15.68
## 123         7.7         2.8          6.7         2.0  virginica      21.56
## 124         6.3         2.7          4.9         1.8  virginica      17.01
## 125         6.7         3.3          5.7         2.1  virginica      22.11
## 126         7.2         3.2          6.0         1.8  virginica      23.04
## 127         6.2         2.8          4.8         1.8  virginica      17.36
## 128         6.1         3.0          4.9         1.8  virginica      18.30
## 129         6.4         2.8          5.6         2.1  virginica      17.92
## 130         7.2         3.0          5.8         1.6  virginica      21.60
## 131         7.4         2.8          6.1         1.9  virginica      20.72
## 132         7.9         3.8          6.4         2.0  virginica      30.02
## 133         6.4         2.8          5.6         2.2  virginica      17.92
## 134         6.3         2.8          5.1         1.5  virginica      17.64
## 135         6.1         2.6          5.6         1.4  virginica      15.86
## 136         7.7         3.0          6.1         2.3  virginica      23.10
## 137         6.3         3.4          5.6         2.4  virginica      21.42
## 138         6.4         3.1          5.5         1.8  virginica      19.84
## 139         6.0         3.0          4.8         1.8  virginica      18.00
## 140         6.9         3.1          5.4         2.1  virginica      21.39
## 141         6.7         3.1          5.6         2.4  virginica      20.77
## 142         6.9         3.1          5.1         2.3  virginica      21.39
## 143         5.8         2.7          5.1         1.9  virginica      15.66
```

```
## 144          6.8         3.2          5.9         2.3  virginica      21.76
## 145          6.7         3.3          5.7         2.5  virginica      22.11
## 146          6.7         3.0          5.2         2.3  virginica      20.10
## 147          6.3         2.5          5.0         1.9  virginica      15.75
## 148          6.5         3.0          5.2         2.0  virginica      19.50
## 149          6.2         3.4          5.4         2.3  virginica      21.08
## 150          5.9         3.0          5.1         1.8  virginica      17.70
```

```
# Group the dataset by species and calculate the mean Sepal.length and Sepal.Wifth for each species
  # df %>% group_by(Species) %>%
  # Store the summarized data in new data frame called iris_summary
  iris_summary = df %>% group_by(Species) %>% summarise(Sepal.Length.Mean = mean(Sepal.Length),Sepal.Wi
```

```
## # A tibble: 3 x 3
##   Species    Sepal.Length.Mean Sepal.Width.Mean
##   <fct>                  <dbl>            <dbl>
## 1 setosa                  5.01             3.43
## 2 versicolor              5.94             2.77
## 3 virginica               6.59             2.97
```

```
  # Export iris_summary in .csv file
  getwd()
```

```
## [1] "A:/CDAC_SM_VITA/5_R_Programming/Day3"
```

```
  setwd("A:/CDAC_SM_VITA/5_R_Programming/Day3")
  write.csv(iris_summary, file = 'Iris_summary.csv')

# Arrange the iris dataset by Sepal,length in desc
  df %>% arrange(desc(Sepal.Length))
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1            7.9         3.8          6.4         2.0  virginica
## 2            7.7         3.8          6.7         2.2  virginica
## 3            7.7         2.6          6.9         2.3  virginica
## 4            7.7         2.8          6.7         2.0  virginica
## 5            7.7         3.0          6.1         2.3  virginica
## 6            7.6         3.0          6.6         2.1  virginica
## 7            7.4         2.8          6.1         1.9  virginica
## 8            7.3         2.9          6.3         1.8  virginica
## 9            7.2         3.6          6.1         2.5  virginica
## 10           7.2         3.2          6.0         1.8  virginica
## 11           7.2         3.0          5.8         1.6  virginica
## 12           7.1         3.0          5.9         2.1  virginica
## 13           7.0         3.2          4.7         1.4 versicolor
## 14           6.9         3.1          4.9         1.5 versicolor
## 15           6.9         3.2          5.7         2.3  virginica
## 16           6.9         3.1          5.4         2.1  virginica
## 17           6.9         3.1          5.1         2.3  virginica
## 18           6.8         2.8          4.8         1.4 versicolor
## 19           6.8         3.0          5.5         2.1  virginica
## 20           6.8         3.2          5.9         2.3  virginica
```

```
## 21          6.7         3.1          4.4         1.4 versicolor
## 22          6.7         3.0          5.0         1.7 versicolor
## 23          6.7         3.1          4.7         1.5 versicolor
## 24          6.7         2.5          5.8         1.8  virginica
## 25          6.7         3.3          5.7         2.1  virginica
## 26          6.7         3.1          5.6         2.4  virginica
## 27          6.7         3.3          5.7         2.5  virginica
## 28          6.7         3.0          5.2         2.3  virginica
## 29          6.6         2.9          4.6         1.3 versicolor
## 30          6.6         3.0          4.4         1.4 versicolor
## 31          6.5         2.8          4.6         1.5 versicolor
## 32          6.5         3.0          5.8         2.2  virginica
## 33          6.5         3.2          5.1         2.0  virginica
## 34          6.5         3.0          5.5         1.8  virginica
## 35          6.5         3.0          5.2         2.0  virginica
## 36          6.4         3.2          4.5         1.5 versicolor
## 37          6.4         2.9          4.3         1.3 versicolor
## 38          6.4         2.7          5.3         1.9  virginica
## 39          6.4         3.2          5.3         2.3  virginica
## 40          6.4         2.8          5.6         2.1  virginica
## 41          6.4         2.8          5.6         2.2  virginica
## 42          6.4         3.1          5.5         1.8  virginica
## 43          6.3         3.3          4.7         1.6 versicolor
## 44          6.3         2.5          4.9         1.5 versicolor
## 45          6.3         2.3          4.4         1.3 versicolor
## 46          6.3         3.3          6.0         2.5  virginica
## 47          6.3         2.9          5.6         1.8  virginica
## 48          6.3         2.7          4.9         1.8  virginica
## 49          6.3         2.8          5.1         1.5  virginica
## 50          6.3         3.4          5.6         2.4  virginica
## 51          6.3         2.5          5.0         1.9  virginica
## 52          6.2         2.2          4.5         1.5 versicolor
## 53          6.2         2.9          4.3         1.3 versicolor
## 54          6.2         2.8          4.8         1.8  virginica
## 55          6.2         3.4          5.4         2.3  virginica
## 56          6.1         2.9          4.7         1.4 versicolor
## 57          6.1         2.8          4.0         1.3 versicolor
## 58          6.1         2.8          4.7         1.2 versicolor
## 59          6.1         3.0          4.6         1.4 versicolor
## 60          6.1         3.0          4.9         1.8  virginica
## 61          6.1         2.6          5.6         1.4  virginica
## 62          6.0         2.2          4.0         1.0 versicolor
## 63          6.0         2.9          4.5         1.5 versicolor
## 64          6.0         2.7          5.1         1.6 versicolor
## 65          6.0         3.4          4.5         1.6 versicolor
## 66          6.0         2.2          5.0         1.5  virginica
## 67          6.0         3.0          4.8         1.8  virginica
## 68          5.9         3.0          4.2         1.5 versicolor
## 69          5.9         3.2          4.8         1.8 versicolor
## 70          5.9         3.0          5.1         1.8  virginica
## 71          5.8         4.0          1.2         0.2     setosa
## 72          5.8         2.7          4.1         1.0 versicolor
## 73          5.8         2.7          3.9         1.2 versicolor
## 74          5.8         2.6          4.0         1.2 versicolor
```

```
## 75          5.8         2.7          5.1         1.9  virginica
## 76          5.8         2.8          5.1         2.4  virginica
## 77          5.8         2.7          5.1         1.9  virginica
## 78          5.7         4.4          1.5         0.4     setosa
## 79          5.7         3.8          1.7         0.3     setosa
## 80          5.7         2.8          4.5         1.3 versicolor
## 81          5.7         2.6          3.5         1.0 versicolor
## 82          5.7         3.0          4.2         1.2 versicolor
## 83          5.7         2.9          4.2         1.3 versicolor
## 84          5.7         2.8          4.1         1.3 versicolor
## 85          5.7         2.5          5.0         2.0  virginica
## 86          5.6         2.9          3.6         1.3 versicolor
## 87          5.6         3.0          4.5         1.5 versicolor
## 88          5.6         2.5          3.9         1.1 versicolor
## 89          5.6         3.0          4.1         1.3 versicolor
## 90          5.6         2.7          4.2         1.3 versicolor
## 91          5.6         2.8          4.9         2.0  virginica
## 92          5.5         4.2          1.4         0.2     setosa
## 93          5.5         3.5          1.3         0.2     setosa
## 94          5.5         2.3          4.0         1.3 versicolor
## 95          5.5         2.4          3.8         1.1 versicolor
## 96          5.5         2.4          3.7         1.0 versicolor
## 97          5.5         2.5          4.0         1.3 versicolor
## 98          5.5         2.6          4.4         1.2 versicolor
## 99          5.4         3.9          1.7         0.4     setosa
## 100         5.4         3.7          1.5         0.2     setosa
## 101         5.4         3.9          1.3         0.4     setosa
## 102         5.4         3.4          1.7         0.2     setosa
## 103         5.4         3.4          1.5         0.4     setosa
## 104         5.4         3.0          4.5         1.5 versicolor
## 105         5.3         3.7          1.5         0.2     setosa
## 106         5.2         3.5          1.5         0.2     setosa
## 107         5.2         3.4          1.4         0.2     setosa
## 108         5.2         4.1          1.5         0.1     setosa
## 109         5.2         2.7          3.9         1.4 versicolor
## 110         5.1         3.5          1.4         0.2     setosa
## 111         5.1         3.5          1.4         0.3     setosa
## 112         5.1         3.8          1.5         0.3     setosa
## 113         5.1         3.7          1.5         0.4     setosa
## 114         5.1         3.3          1.7         0.5     setosa
## 115         5.1         3.4          1.5         0.2     setosa
## 116         5.1         3.8          1.9         0.4     setosa
## 117         5.1         3.8          1.6         0.2     setosa
## 118         5.1         2.5          3.0         1.1 versicolor
## 119         5.0         3.6          1.4         0.2     setosa
## 120         5.0         3.4          1.5         0.2     setosa
## 121         5.0         3.0          1.6         0.2     setosa
## 122         5.0         3.4          1.6         0.4     setosa
## 123         5.0         3.2          1.2         0.2     setosa
## 124         5.0         3.5          1.3         0.3     setosa
## 125         5.0         3.5          1.6         0.6     setosa
## 126         5.0         3.3          1.4         0.2     setosa
## 127         5.0         2.0          3.5         1.0 versicolor
## 128         5.0         2.3          3.3         1.0 versicolor
```

```
## 129          4.9         3.0          1.4         0.2     setosa
## 130          4.9         3.1          1.5         0.1     setosa
## 131          4.9         3.1          1.5         0.2     setosa
## 132          4.9         3.6          1.4         0.1     setosa
## 133          4.9         2.4          3.3         1.0 versicolor
## 134          4.9         2.5          4.5         1.7  virginica
## 135          4.8         3.4          1.6         0.2     setosa
## 136          4.8         3.0          1.4         0.1     setosa
## 137          4.8         3.4          1.9         0.2     setosa
## 138          4.8         3.1          1.6         0.2     setosa
## 139          4.8         3.0          1.4         0.3     setosa
## 140          4.7         3.2          1.3         0.2     setosa
## 141          4.7         3.2          1.6         0.2     setosa
## 142          4.6         3.1          1.5         0.2     setosa
## 143          4.6         3.4          1.4         0.3     setosa
## 144          4.6         3.6          1.0         0.2     setosa
## 145          4.6         3.2          1.4         0.2     setosa
## 146          4.5         2.3          1.3         0.3     setosa
## 147          4.4         2.9          1.4         0.2     setosa
## 148          4.4         3.0          1.3         0.2     setosa
## 149          4.4         3.2          1.3         0.2     setosa
## 150          4.3         3.0          1.1         0.1     setosa
```

```
# Find the unique species present in the dataset
unique(df['Species'])
```

```
##        Species
## 1       setosa
## 51  versicolor
## 101  virginica
```

```
### DATASET 2
# 1) Load the mtcars dataset and display first 7 rows
df = mtcars; head(df)
```

```
##                    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
```

```
# 2) filter the dataset to show only cars with mpg greater than 20
df %>% filter(mpg >= 20)
```

```
##                mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4      21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710     22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Merc 240D      24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
```

```
## Merc 230       22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona  21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Fiat X1-9      27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Volvo 142E     21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
# 3) create a new data frame called selected_cars by selecting mpg,hp & cyl columns
selected_cars = df[c('mpg', 'hp', 'cyl')]; selected_cars
```

```
##                      mpg  hp cyl
## Mazda RX4           21.0 110   6
## Mazda RX4 Wag       21.0 110   6
## Datsun 710          22.8  93   4
## Hornet 4 Drive      21.4 110   6
## Hornet Sportabout   18.7 175   8
## Valiant             18.1 105   6
## Duster 360          14.3 245   8
## Merc 240D           24.4  62   4
## Merc 230            22.8  95   4
## Merc 280            19.2 123   6
## Merc 280C           17.8 123   6
## Merc 450SE          16.4 180   8
## Merc 450SL          17.3 180   8
## Merc 450SLC         15.2 180   8
## Cadillac Fleetwood  10.4 205   8
## Lincoln Continental 10.4 215   8
## Chrysler Imperial   14.7 230   8
## Fiat 128            32.4  66   4
## Honda Civic         30.4  52   4
## Toyota Corolla      33.9  65   4
## Toyota Corona       21.5  97   4
## Dodge Challenger    15.5 150   8
## AMC Javelin         15.2 150   8
## Camaro Z28          13.3 245   8
## Pontiac Firebird    19.2 175   8
## Fiat X1-9           27.3  66   4
## Porsche 914-2       26.0  91   4
## Lotus Europa        30.4 113   4
## Ford Pantera L      15.8 264   8
## Ferrari Dino        19.7 175   6
## Maserati Bora       15.0 335   8
## Volvo 142E          21.4 109   4
```

```
# 4) create a new column hp_per_cyl that calculates horsepower per cylinder
df %>% mutate(
  hp_per_cyl = hp / cyl
)
```

```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
```

```
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
##                     hp_per_cyl
## Mazda RX4             18.33333
## Mazda RX4 Wag         18.33333
## Datsun 710            23.25000
## Hornet 4 Drive        18.33333
## Hornet Sportabout     21.87500
## Valiant               17.50000
## Duster 360            30.62500
## Merc 240D             15.50000
## Merc 230              23.75000
## Merc 280              20.50000
## Merc 280C             20.50000
## Merc 450SE            22.50000
## Merc 450SL            22.50000
## Merc 450SLC           22.50000
## Cadillac Fleetwood    25.62500
## Lincoln Continental   26.87500
## Chrysler Imperial     28.75000
## Fiat 128              16.50000
## Honda Civic           13.00000
## Toyota Corolla        16.25000
## Toyota Corona         24.25000
```

```
## Dodge Challenger      18.75000
## AMC Javelin           18.75000
## Camaro Z28            30.62500
## Pontiac Firebird      21.87500
## Fiat X1-9             16.50000
## Porsche 914-2         22.75000
## Lotus Europa          28.25000
## Ford Pantera L        33.00000
## Ferrari Dino          29.16667
## Maserati Bora         41.87500
## Volvo 142E            27.25000
```

```
# 5) group the dataset by cyl and calculate the average mpg & hp for each group
df %>% group_by(cyl) %>% mutate(AverageMPG = mean(mpg), AverageHP = mean(hp) ) %>% ungroup()
```

```
## # A tibble: 32 x 13
##      mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb AverageMPG
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>      <dbl>
##  1  21       6  160    110  3.9   2.62  16.5     0     1     4     4       19.7
##  2  21       6  160    110  3.9   2.88  17.0     0     1     4     4       19.7
##  3  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1       26.7
##  4  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1       19.7
##  5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2       15.1
##  6  18.1     6  225    105  2.76  3.46  20.2     1     0     3     1       19.7
##  7  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4       15.1
##  8  24.4     4  147.    62  3.69  3.19  20       1     0     4     2       26.7
##  9  22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2       26.7
## 10  19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4       19.7
## # i 22 more rows
## # i 1 more variable: AverageHP <dbl>
```

```
df %>% group_by(cyl) %>% summarise(AverageMPG = mean(mpg), AverageHP = mean(hp)) %>% ungroup()
```

```
## # A tibble: 3 x 3
##     cyl AverageMPG AverageHP
##   <dbl>      <dbl>     <dbl>
## 1     4       26.7      82.6
## 2     6       19.7     122.
## 3     8       15.1     209.
```

```
# 6) count the number of cars for each number of cylinder
df %>% group_by(cyl, ) %>% count()
```

```
## # A tibble: 3 x 2
## # Groups:   cyl [3]
##     cyl     n
##   <dbl> <int>
## 1     4    11
## 2     6     7
## 3     8    14
```

```r
# 7) arrange the dataset by mpg in descending order and export this
df1 = df %>% arrange(desc('mpg')); df1
```

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

```r
  # dataframe in .txt file with separator as "$"
  write.table(df1,file = '7.txt', sep ='$' ); df1
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
```

```
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
# 8) group the dataset by gear and calculate the total number of cars and average mpg

# 9) create a new column "Performance" that categorise cars based on their mpg into low, medium and hig
?case_when

df %>% mutate(Performance = case_when(
  mpg < 20 ~ 'Low',
  mpg >= 20 & mpg < 25 ~ 'Medium',
  mpg >= 25 ~ 'High'
))
```

```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4          21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag      21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710         22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive     21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant            18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360         14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D          24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230           22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280           19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C          17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE         16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL         17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC        15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4  8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128           32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic        30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla     33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
```

```
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
##                     Performance
## Mazda RX4                Medium
## Mazda RX4 Wag            Medium
## Datsun 710               Medium
## Hornet 4 Drive           Medium
## Hornet Sportabout           Low
## Valiant                     Low
## Duster 360                  Low
## Merc 240D                Medium
## Merc 230                 Medium
## Merc 280                    Low
## Merc 280C                   Low
## Merc 450SE                  Low
## Merc 450SL                  Low
## Merc 450SLC                 Low
## Cadillac Fleetwood          Low
## Lincoln Continental         Low
## Chrysler Imperial           Low
## Fiat 128                   High
## Honda Civic                High
## Toyota Corolla             High
## Toyota Corona            Medium
## Dodge Challenger            Low
## AMC Javelin                 Low
## Camaro Z28                  Low
## Pontiac Firebird            Low
## Fiat X1-9                  High
## Porsche 914-2              High
## Lotus Europa               High
## Ford Pantera L              Low
## Ferrari Dino                Low
## Maserati Bora               Low
## Volvo 142E               Medium
```

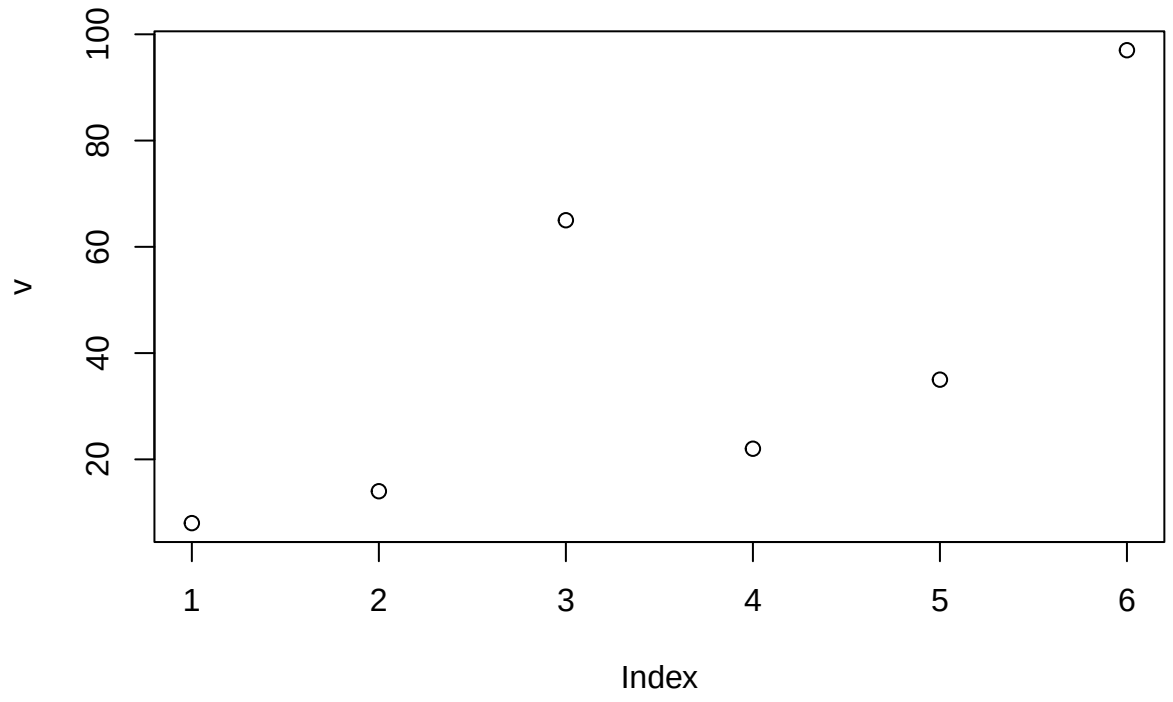
```
# 10) introduce some NA values into the hp column, calculate the total number of missing values and the
# 11) filter the dataset to show cars with mpg greater than 20 and hp less than 100
df
```

```
##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
```

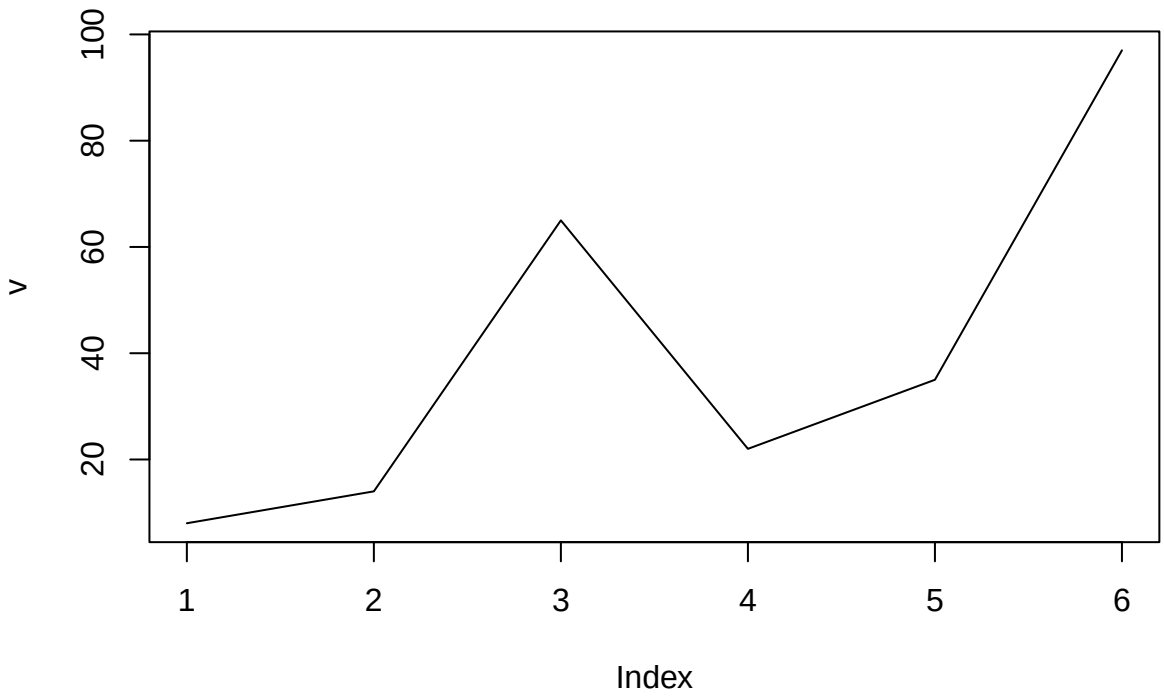
```
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3  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

```
#### LINE PLOT ####

data()
?plot
# type
# what type of plot should be drawn. Possible types are
#
# "p" for points,
#
# "l" for lines,
#
# "b" for both,
#
# "c" for the lines part alone of "b",
#
# "o" for both ‘overplotted’,
#
# "h" for ‘histogram’ like (or ‘high-density’) vertical lines,
#
# "s" for stair steps,
#
# "S" for other steps, see ‘Details’ below,
#
# "n" for no plotting.
```

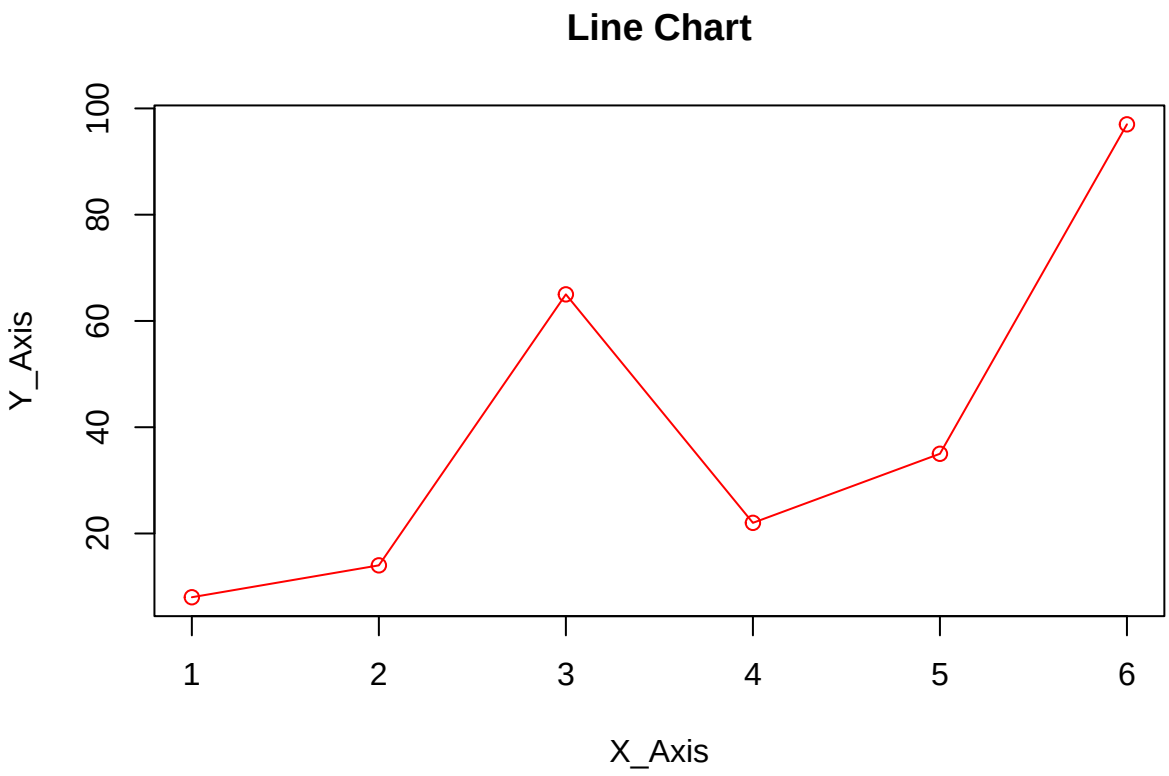
```
?iris

v = c(8,14,65,22,35,97)
plot(v, type = 'p')
```



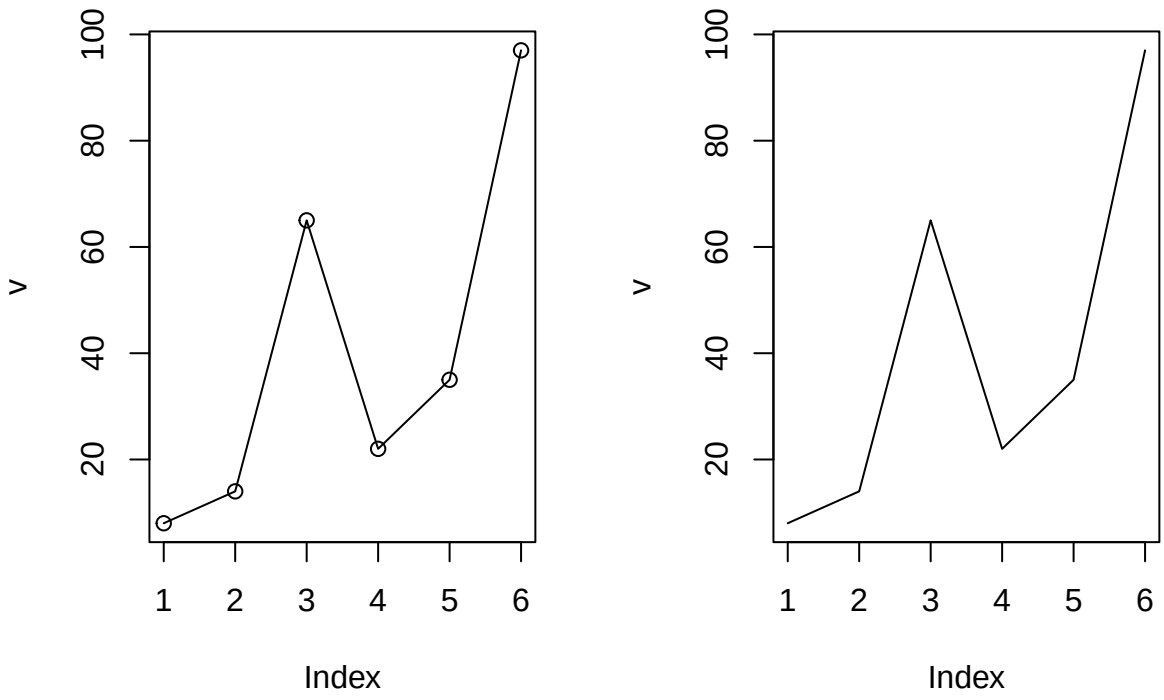

```
plot(v, type='l')
```



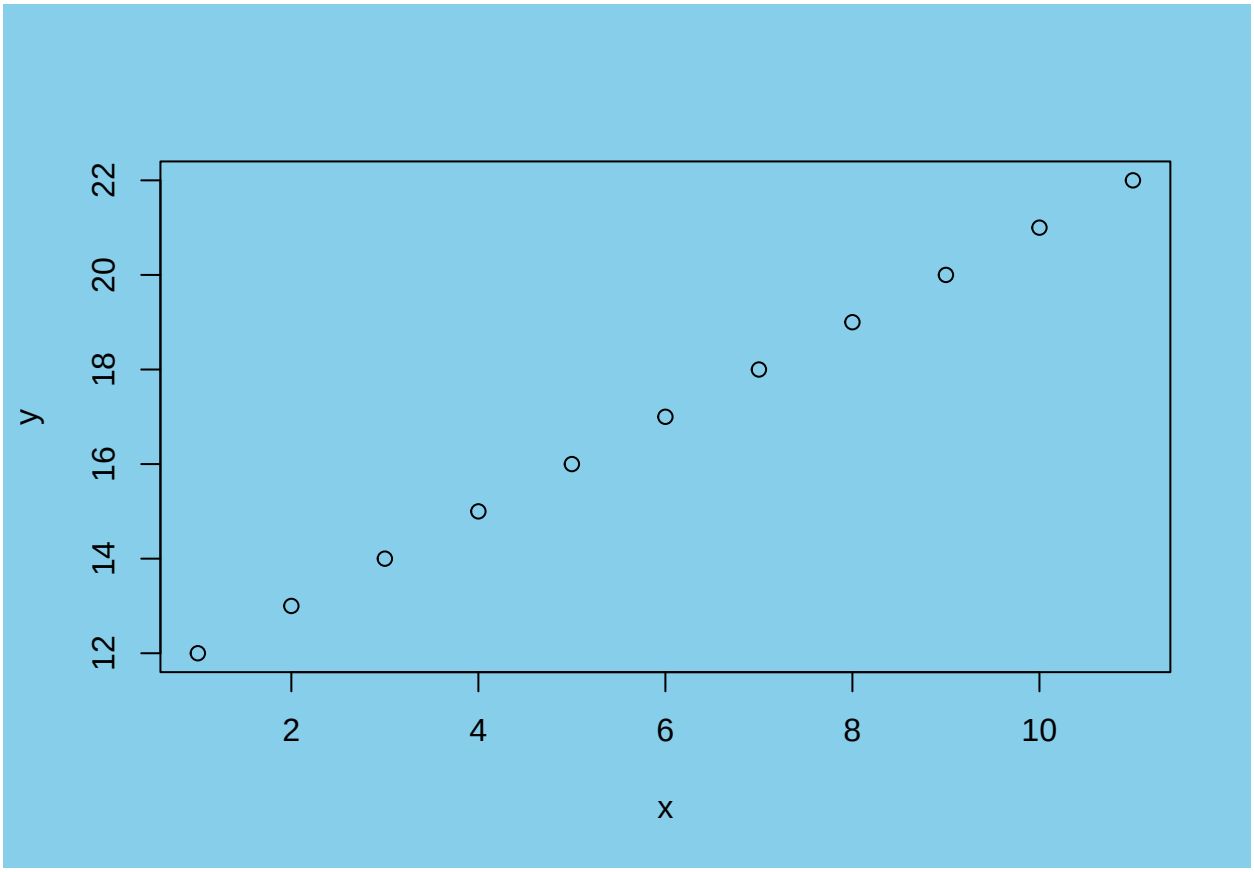

```
#plotting detailed chart
plot(v, type = 'o', col = 'red', xlab='X_Axis', ylab='Y_Axis' ,main = 'Line Chart')
```




```
### PAR - Partition for multiple partitions
par(mfrow = c(1,2))
plot(v, type = 'o')
plot(v, type = 'l')
```




```
# resetting the frame back to 1 image
par(mfrow = c(1, 1))

# to change the background color of the plots
par(bg = 'skyblue')
x = c(1:11)
y = c(12:22)

# plotting 2 lines in one single plot
plot(x, y)
```



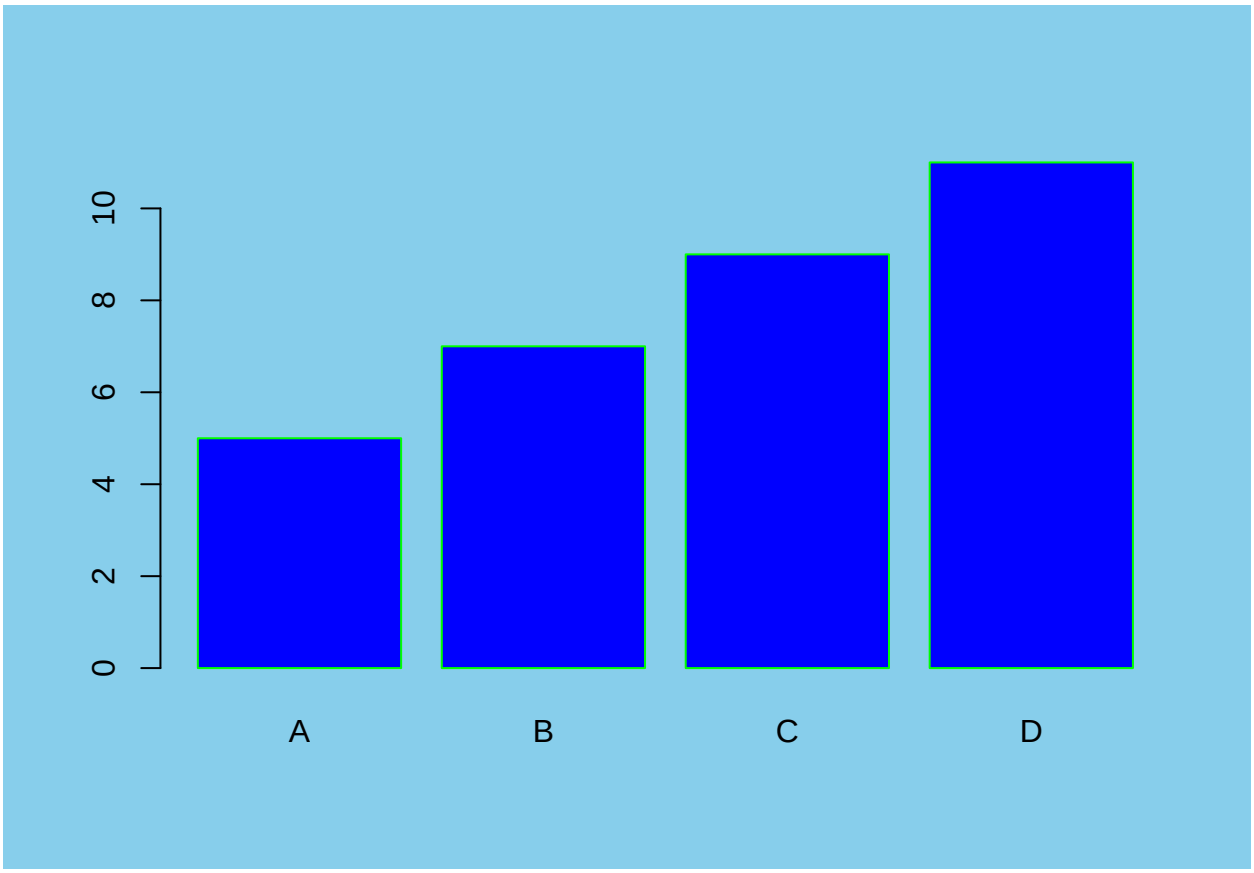

```
data = iris
# names of column
names(data)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

```
View(data)
#Structure of data
str(data)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

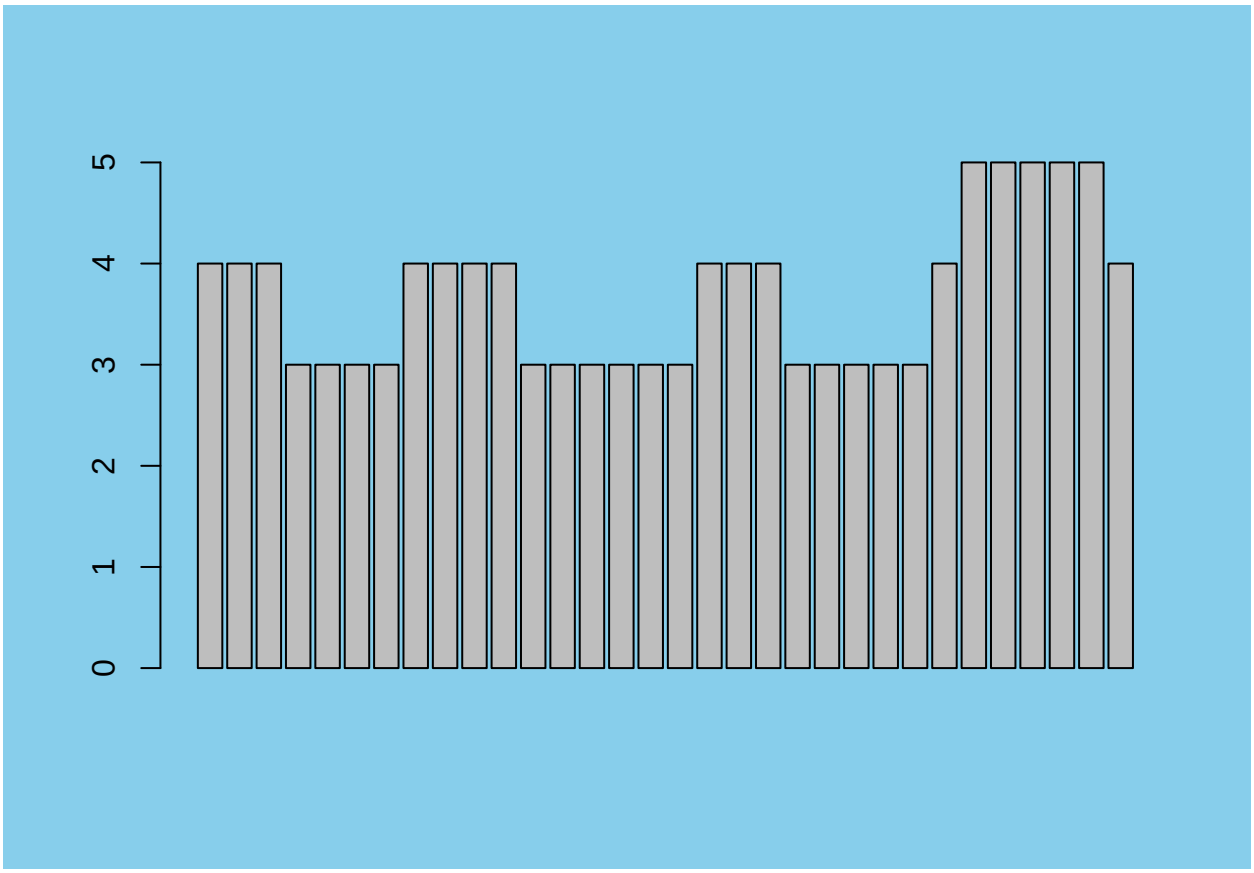
```
### Bar Plot
?barplot
H = c(5,7,9,11)
M = c('A', 'B', 'C', 'D')
# barplot
barplot(H, names.arg= M, col = 'blue', border = 'green')
```




```
#mtcars data
counts = table(mtcars$gear);counts
```

```
##
##  3  4  5
## 15 12  5
```

```
barplot(mtcars$gear)
```




```
mtcars
```

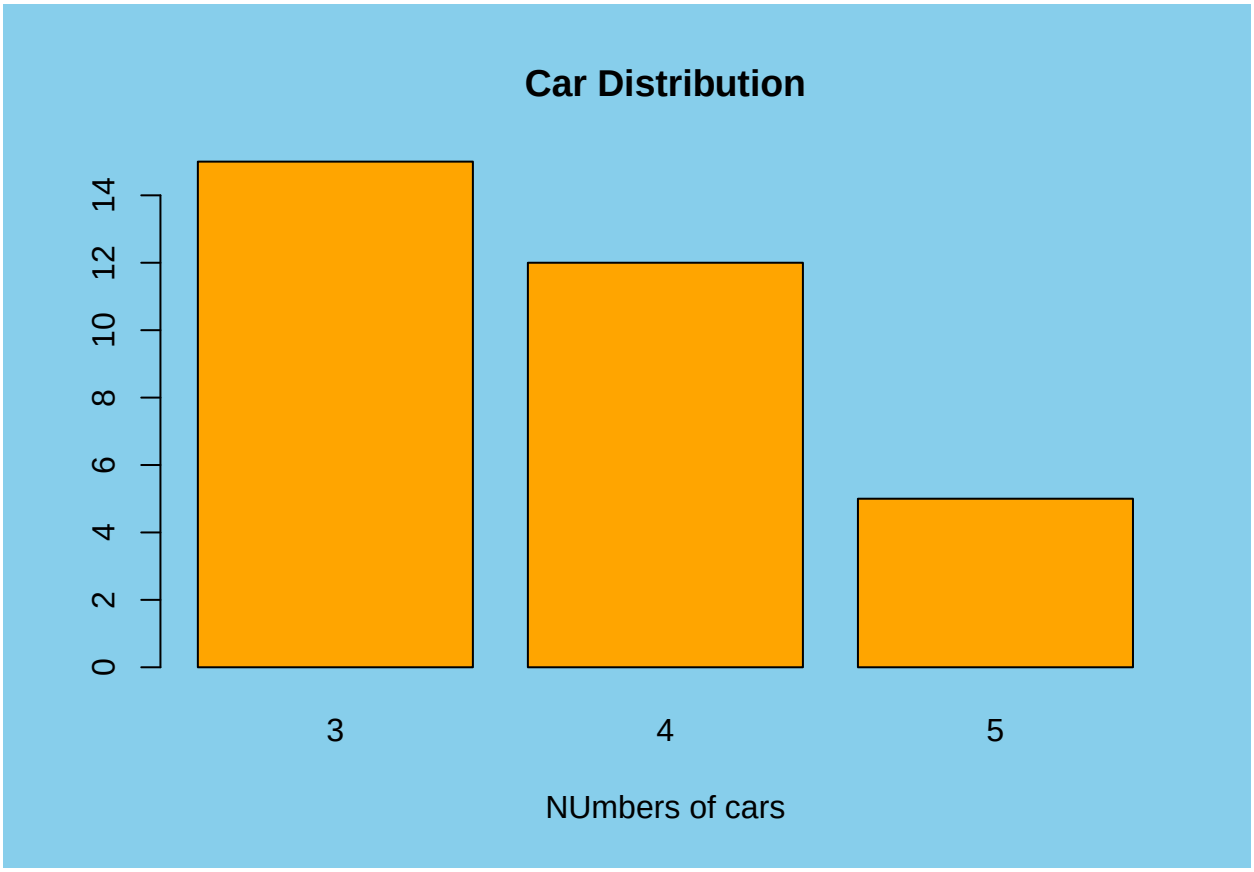
```
##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C         17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE        16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL        17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC       15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128          32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic       30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla    33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona     21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger  15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin       15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
```

```
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
mtcars$gear
```

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

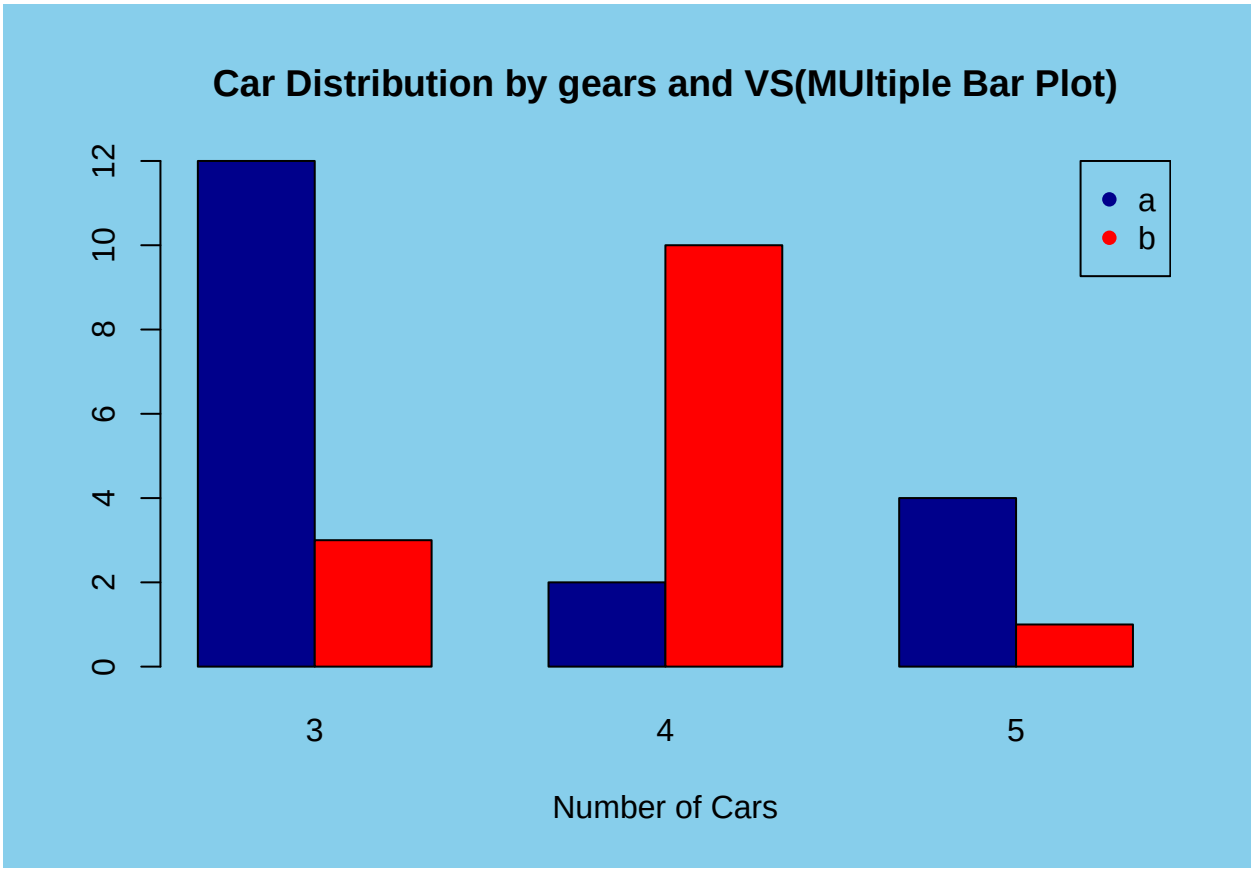
```
barplot(counts, main="Car Distribution", xlab='NUmbers of cars', col='orange')
```




```
### Multiple barPlot

counts = table(mtcars$vs, mtcars$gear)
rownames(counts)
```

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

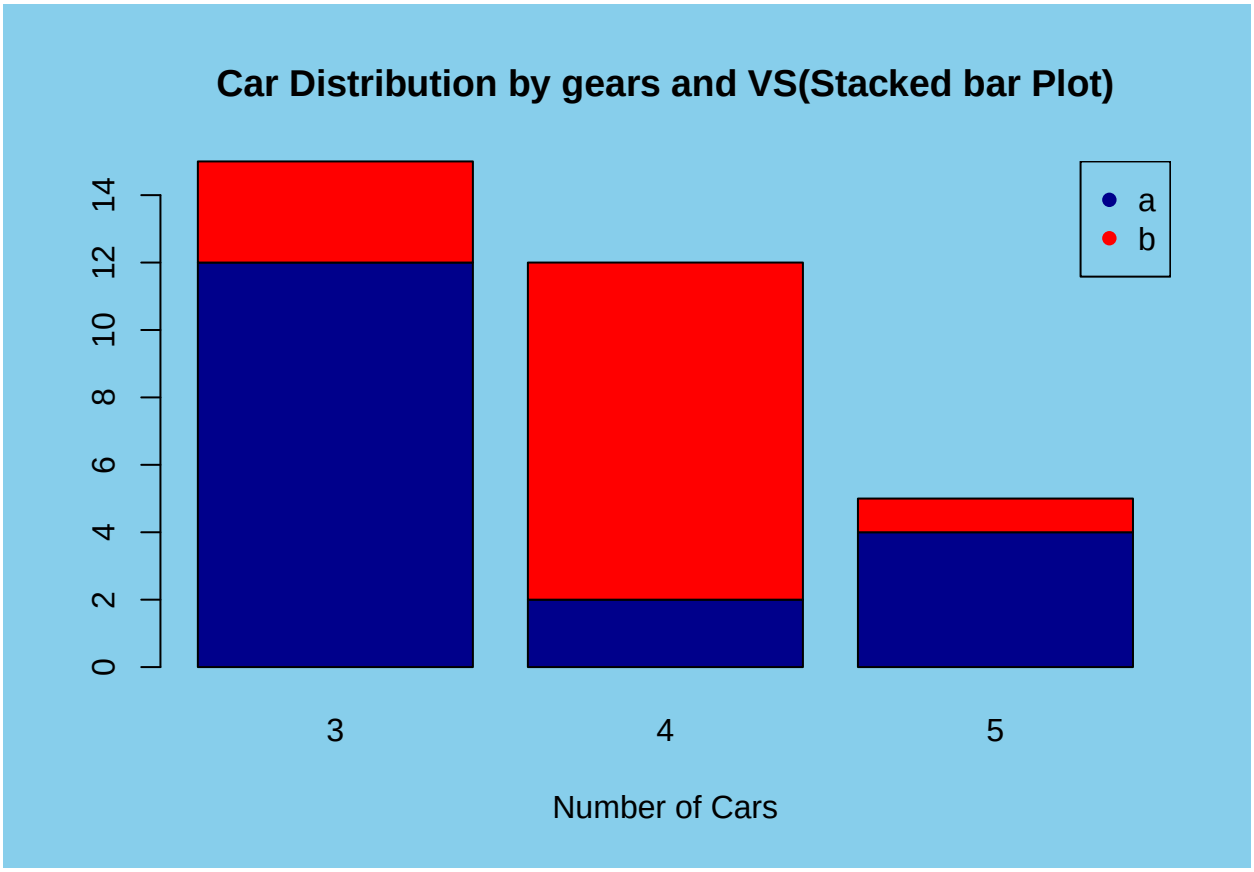
```
barplot(counts, main="Car Distribution by gears and VS(MUltiple Bar Plot)", xlab = 'Number of Cars',
        col=c('darkblue', 'red'), beside=TRUE)
legend('topright', pch = 16, col=c('darkblue', 'red'), c('a', 'b'))
```



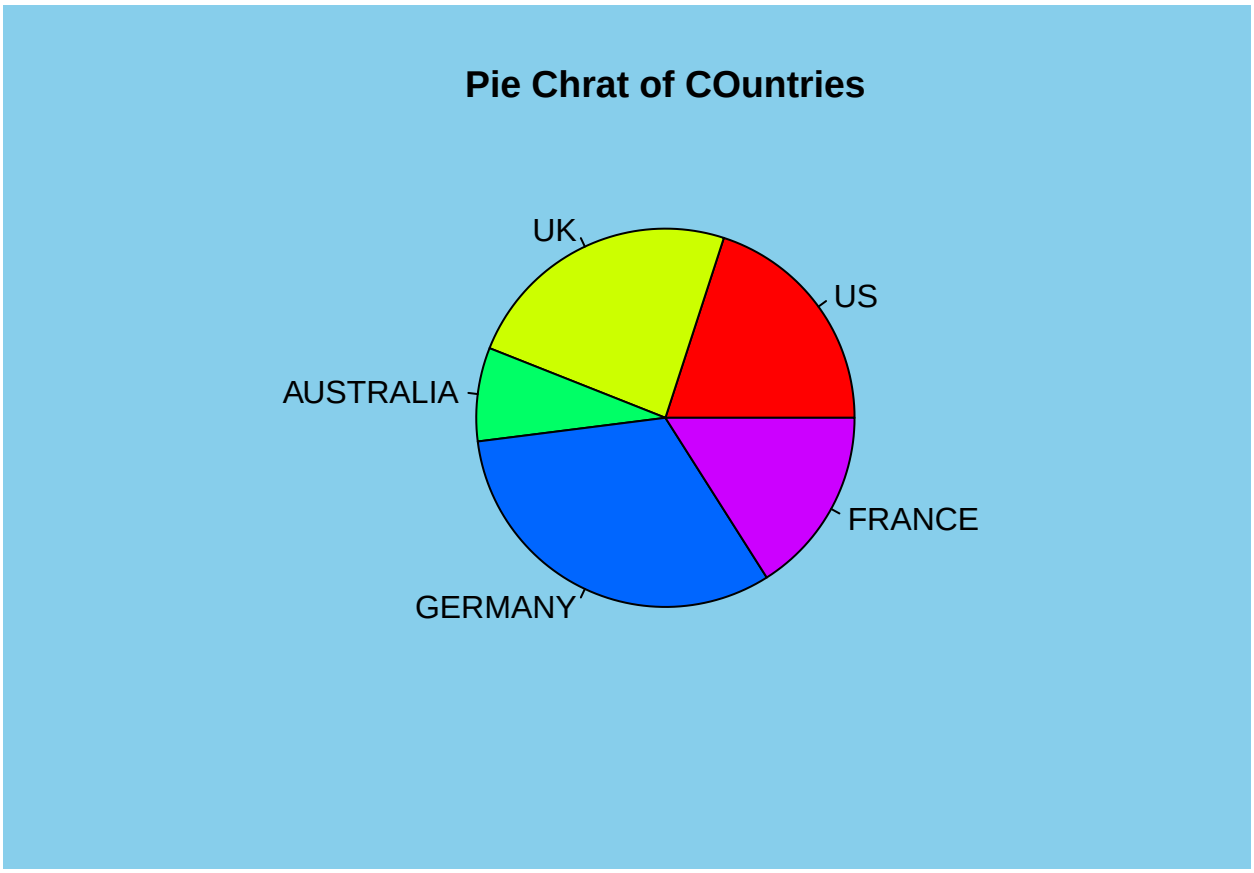

```
### Stacked barplot

counts = table(mtcars$vs, mtcars$gear)
rownames(counts)
```

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

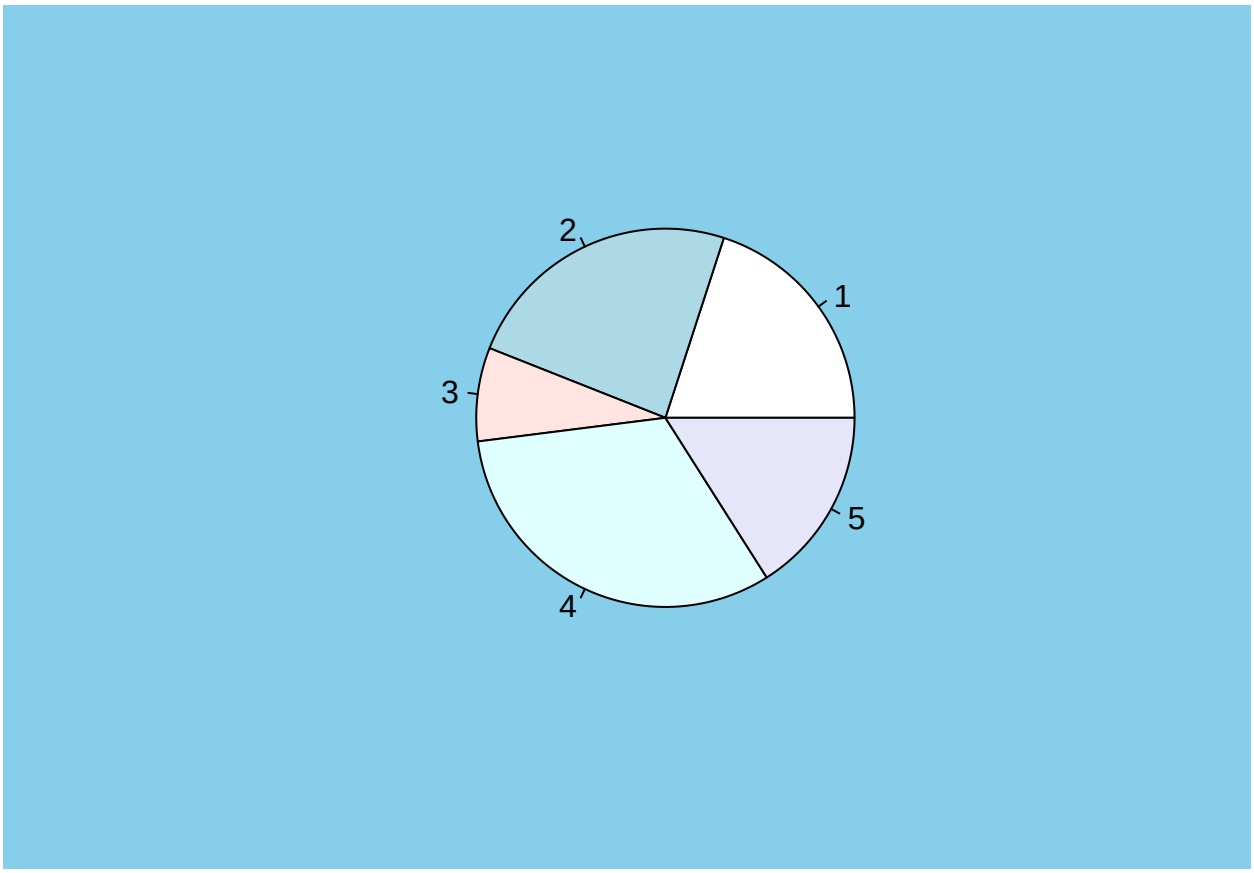
```
barplot(counts, main="Car Distribution by gears and VS(Stacked bar Plot)", xlab = 'Number of Cars',
        col=c('darkblue', 'red'), beside=FALSE)
legend('topright', pch = 16, col=c('darkblue', 'red'), c('a', 'b'))
```



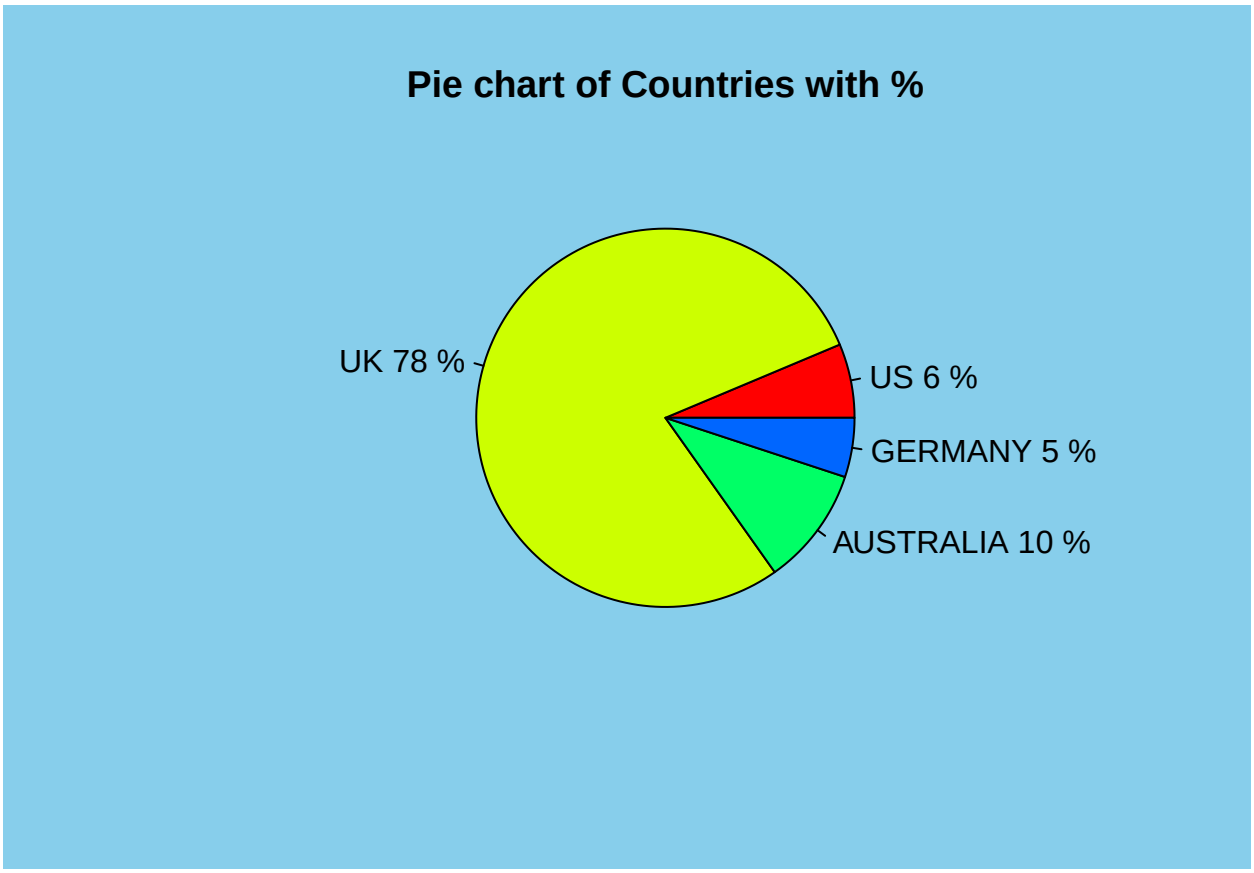

```
### Pie Chart
?paste
?round
?pie
slices = c(10,12,4,16,8)
label = c('US', 'UK', 'AUSTRALIA', 'GERMANY', 'FRANCE')
pie(slices, labels = label, main = 'Pie Chrat of COuntries', col = rainbow(5))
```




```
pie(slices)
```




```r
# pie chart with percentage written on it
slices = c(10,124,16,8)
lable = c('US', 'UK', 'AUSTRALIA', 'GERMANY', 'FRANCE')
pct = round(slices / sum(slices) * 100)
lable = paste(lable, pct)
lable = paste(lable, '%', sep=' ')
pie(slices, labels = lable, col = rainbow(length(lable)), main = 'Pie chart of Countries with %')
```


Pie chart of Countries with %
UK 78 %
US 6 %
GERMANY 5 %
AUSTRALIA 10 %