

**Vector.R**

```

#=====
# Vectors.R
#=====

#-----
# Use # at the beginning of line to create a comment.
#-----

#-----
# Creating a Variables
#-----
x <- 7
y <- 10
z <- 'Hello'

a <- x + y
b <- x - y
c <- x * y
d <- x / y          # this create a float it does not floor or ceil the value

# e <- x + z          # be careful of variable types and operations

e <- x^2             # exponent notation
e                      # output e

f <- (a + d*(3+ c))^3 / (x) # complex math statement
f                      # output f

rm(list=ls())        # this deletes all variables - housekeeping code

#-----
# Working with Vectors
#-----
# creating a vector
prices <- c(2.32, 4.21, 0.21, 5.02, 7)
first.names <- c('Bill', 'Sue', 'Jeff')
mix.values <- c(3.2, 'Bill')

# getting and setting values of a vector
prices[1]              # getting a single value
prices[3]

prices[2:4]             # getting a range of values

prices                 # output prices to the console
prices[1] <- 0.25       # assign a value to first element of vector
prices

```

```
prices[2:3] <- 0.10      # assign a value to range of elements of
prices

prices <- prices[-4]     # deleting 4th element in the vector
prices

prices <- c(prices, 3.25) # appending element to the end of the vector
prices
```

```
#-----
# Useful functions for Vectors
#-----
```

```
len <- length(prices)      # get the length of a vector
sprintf("Prices has %d elements", len ) # output format string

mean(prices)               # calculate the mean of prices vector

sprintf("Prices has mean of:%f", mean(prices) )
sprintf("Prices has mean of:%.2f", mean(prices) ) # showing two digits

std.dev <- sd(prices)      # calculate the std.dev of prices
vector

sprintf("The standard deviation of Prices:%.3f", std.dev )

hist(prices)              # create a histogram

pattern <- rep(c(1, 2, 3, 3, 3, 4, 5,5), times = 4) #repeat the complete vector
pattern
hist(pattern)
```

```
sequence <- seq(from = 0.0, to = 2.0, by = 0.1) # create a sequence vector
sequence
```

```
any(prices > 2.00)      # reports whether any of those values are TRUE

all(prices > 2.00)      # reports if all the values are TRUE

anyNA(prices)          # check if any element of prices is NA

any(is.na(prices))     # check if any element of prices is NA
```

```
#-----
# Stock data with Vectors
#-----
```

```
appl.close <-c(172.26, 172.23, 173.03, 175.00, 174.35, 174.33, 174.29, 175.28,
               177.09, 176.19, 179.10, 179.26, 178.46, 177.00, 177.04, 174.22,
               171.11, 171.51, 167.96, 166.97, 167.43, 167.78, 160.50, 156.49,
```

163.03)

```
appl.vol <- c(25555900, 29517900, 22434600, 23660000, 20567800, 21584000,
23959900,
              18667700, 25418100, 29565900, 34386800, 31193400, 32425100,
27108600,
              32689100, 51105100, 41529000, 39143000, 50640400, 46048200,
32478900,
              47230800, 86593800, 72738500, 68056200)
```

*# plotting data*

```
par(mfrow=c(2,1))
plot(appl.close, type = 'l')
plot(appl.vol, type = 'l')
```

*# plot histogram*

```
hist(appl.close)
hist(appl.vol)
```

*# calculate close stats*

```
close.mean <- mean(appl.close)
close.sd <- sd(appl.close)
close.range <- max(appl.close) - min(appl.close)
```

*# calculate vol stats*

```
vol.mean <- mean(appl.vol)
vol.sd <- sd(appl.vol)
vol.range <- max(appl.vol) - min(appl.vol)
```

*# Create simple report*

```
sprintf("AAPL closing mean:%0.2f standard deviation:%0.4f and range:%0.2f",
        close.mean, close.sd, close.range)
```

```
sprintf("AAPL volume mean:%d standard deviation:%.0f and range:%d",
        vol.mean, vol.sd, vol.range)
```

**Project Code**

```

#=====
# Vector Project
#=====

#=====
# Project Coding
#=====

appl.close <-c(172.26, 172.23, 173.03, 175.00, 174.35, 174.33, 174.29, 175.28,
              177.09, 176.19, 179.10, 179.26, 178.46, 177.00, 177.04, 174.22,
              171.11, 171.51, 167.96, 166.97, 167.43, 167.78, 160.50, 156.49,
              163.03)

appl.vol <- c(25555900, 29517900, 22434600, 23660000, 20567800, 21584000, 23959900,
             18667700, 25418100, 29565900, 34386800, 31193400, 32425100, 27108600,
             32689100, 51105100, 41529000, 39143000, 50640400, 46048200, 32478900,
             47230800, 86593800, 72738500, 68056200)

#=====

#-----
# Student Code
#-----
# create a variable named x and store 10 in it
# hint code:
x <- "PUT SOME CODE HERE"

#-----
# Student Code
#-----
# square x and store it in y
# hint code:
"PUT SOME CODE HERE" <- x"PUT SOME CODE HERE"

#-----
# Student Code
#-----
# output the value of y to the concole
y

#-----
# Student Code
#-----
# plotting appl.close and appl.vol
par(mfrow=c(2,1))
plot("PUT SOME CODE HERE", type = 'l')
plot("PUT SOME CODE HERE", type = 'l')

```

```
par(mfrow=c(2,1))
plot(appl.close, type = 'l')
plot(appl.vol, type = 'l')
```

```
#-----
# Student Code
#-----
```

```
# calculate close stats mean, standard error and range
mean("PUT SOME CODE HERE")
```

```
"PUT SOME CODE HERE"(appl.close)
```

```
max("PUT SOME CODE HERE") - "PUT SOME CODE HERE"(appl.close)
```

```
mean(appl.close)
sd(appl.close)
max(appl.close) - min(appl.close)
```

```
#-----
# Student Code
#-----
```

```
# creating a vector price with data 2.32, 4.21, 0.21, 5.02, 7
```

```
"PUT SOME CODE HERE" <- "PUT SOME CODE HERE"(2.32, 4.21, 0.21, 5.02, 7)
```

```
prices <- c(2.32, 4.21, 0.21, 5.02, 7)
```

```
#-----
# Student Code
#-----
```

```
# getting first and third values of a vector and output prices to the console
prices["PUT SOME CODE HERE"]
prices["PUT SOME CODE HERE"]
```

```
#-----
# Student Code
#-----
```

```
# getting element 2-4 for the prices and output values to the console
```

```
prices["PUT SOME CODE HERE"]
```

```
#-----
# Student Code
#-----
```

```
# set first prices element to 0.25 and output values to the console
prices["PUT SOME CODE HERE"] <- "PUT SOME CODE HERE"
```

#-----  
# *End*  
#-----



# **Project Code Solution**

```
#=====
# Vector Project
#=====
```

```
#=====
# Project Coding
#=====
```

```
appl.close <-c(172.26, 172.23, 173.03, 175.00, 174.35, 174.33, 174.29, 175.28,
              177.09, 176.19, 179.10, 179.26, 178.46, 177.00, 177.04, 174.22,
              171.11, 171.51, 167.96, 166.97, 167.43, 167.78, 160.50, 156.49,
              163.03)
```

```
appl.vol <- c(25555900, 29517900, 22434600, 23660000, 20567800, 21584000, 23959900,
             18667700, 25418100, 29565900, 34386800, 31193400, 32425100, 27108600,
             32689100, 51105100, 41529000, 39143000, 50640400, 46048200, 32478900,
             47230800, 86593800, 72738500, 68056200)
```

```
#=====
```

```
#-----
# Student Code
#-----
# create a variable named x and store 10 in it
# hint code: x <- "PUT SOME CODE HERE"
```

```
x <- 10
```

```
#-----
# Student Code
#-----
# square x and store it in y
# hint code: "PUT SOME CODE HERE" <- x"PUT SOME CODE HERE"
```

```
y <- x^2
```

```
#-----
# Student Code
#-----
# output the value of y to the concole
y
```

```
#-----
# Student Code
#-----
# plotting appl.close and appl.vol
par(mfrow=c(2,1))
# plot("PUT SOME CODE HERE", type = 'l')
# plot("PUT SOME CODE HERE", type = 'l')
```

```
par(mfrow=c(2,1))
plot(appl.close, type = 'l')
plot(appl.vol, type = 'l')
```

```
#-----
# Student Code
#-----
# calculate close stats mean, standard error and range
# mean("PUT SOME CODE HERE")
# "PUT SOME CODE HERE"(appl.close)
# max("PUT SOME CODE HERE") - "PUT SOME CODE HERE"(appl.close)
```

```
mean(appl.close)
sd(appl.close)
max(appl.close) - min(appl.close)
```

```
#-----
# Student Code
#-----
# creating a vector price with data 2.32, 4.21, 0.21, 5.02, 7

# "PUT SOME CODE HERE" <- "PUT SOME CODE HERE"(2.32, 4.21, 0.21, 5.02, 7)
```

```
prices <- c(2.32, 4.21, 0.21, 5.02, 7)
```

```
#-----
# Student Code
#-----
# getting first and third values of a vector and output prices to the console
# prices["PUT SOME CODE HERE"]
# prices["PUT SOME CODE HERE"]
```

```
prices[1]
prices[3]
prices
```

```
#-----
# Student Code
#-----
# getting element 2-4 for the prices and output values to the console

# prices["PUT SOME CODE HERE"]
```

```
prices[2:4]          # getting a range of values
```

```
#-----
# Student Code
#-----
```

```
# set first prices element to 0.25 and output values to the console  
# prices["PUT SOME CODE HERE"] <- "PUT SOME CODE HERE"
```

```
prices[1] <- 0.25          # assign a value to first element of vector  
prices
```