Section 3.1

Exercise 1: Guess the result of each of the following results will be, then check answers

a) 4 + 2 * 8 =	My Guess: 20	Actual: 20
b) 4 + 2 * 8 + 3 =	My Guess: 23	Actual: 23
c) -2^2 =	My Guess: -4	Actual: -4
d) 1 + 2^2 * 4 =	My Guess: 17	Actual: 17
e) (2 + 4) / 3 / 2 =	My Guess: 1	Actual: 1

Section 3.2

Exercise 2: Guess the result of the following, then check answers

a) 5 / 0 =	My Guess: Inf	Actual: Inf
b) 1 / Inf =	My Guess: 0	Actual: 0
c) 0 / 0 =	My Guess: NaN	Actual: NaN
d) Inf + 1 =	My Guess: Inf	Actual: Inf

Section 3.3

Exercise 3: What variable type is stored, then check answers

a) x <- 45	My Guess: Double	Actual: Double
b) x <- "foo"	My Guess: Character	Actual: Character
c) x <- FALSE	My Guess: Logical	Actual: Logical
d) x <- NULL	My Guess: NULL	Actual: NULL

Exercise 4: Guess final value of x in the following sequence of commands, then check answer

```
x <- 2                  # x = 2
x <- x * 2 + 1          # x = 5
x <- x * 3              # x = 15
x                       # x = 15
                        My Guess: x = 15        Actual: x = 15
```

Exercise 5: Write commands that do the following (in order):

1. Create a variable y containing the value 5
2. Overwrite the value of y by the value 3 * y
3. Copy the calue of y into a new variable z

```
y <- 5          # y = 5
y <- 3 * y      # y = 15
z <- y          # z = y = 15
```

Section 3.4

Exercise 6: Besides sqrt(), what other R function is described on the help page?
The other R function described on the help page is for abs().

Exercise 7: From the help page, how many arguments does signif() have?
There are two arguments needed for signif() in R.

Exercise 8: Answer the following from the help page for signif()
a) What is the default value for the digits argument?
The default argument for signif is 6 digits.

b) To how many significant digits will the value 342.88937 be printed by the command,
"signif(x = 342.88937)?
It will print 342.889.

Exercise 9: Do the following
a) Write a command using named argument matching that prints the value 342.88937 to 5 significant digits.

```
signif(x = 342.88937, digits = 5)
```

b) Write a command using positional matching that prints 342.88937 to 5 significant digits.

```
signif(342.88937, 5)
```

Section 3.5
Exercise 10: Create a few variables named x, y, and z. Then type the following sequence of commands. What are the outputs from the three calls to ls()?

```
ls()                    # output: "x" "y" "z"
rm(x)
ls()                    # output: "y" "z"
rm(list = ls())
ls()                    # output: character(0)
```

Section 3.6
Exercise 11: Write a command using c() that creates a vector containing the values: 3, 7, 2, 8

```
var <- c(3, 7, 2, 8)
```

Exercise 12: Write a command using matrix() that creates the following matrix: ((2, 6), (4, 8))

```
mat <- matrix(c(2, 4, 6, 8), nrow = 2, ncol = 2)
```

Exercise 13: Write a command using list() that creates a list containing the elements: "e", 9, TRUE

```
list1 <- list("e", 9, TRUE)
```

Exercise 14: Write a command using data.frame() that creates the following data set:

Category	Value
A	5
A	4
B	6
B	6
C	9
C	8

```
df <- data.frame(Category = c("A", "A", "B", "B", "C", "C"), Value = c(5, 4, 6, 6,
9, 8))
```

Section 4.2

Exercise 15: Guess what the result of each of the following will be,
then check your answers

```
a) x <- c(2, 3, 4, 5)
   y <- c(6, 7, 8, 9)
   c(x, y)

                        My Guess: 2 3 4 5 6 7 8 9      Actual: 2 3 4 5 6 7 8 9

b)  x <- c(2, 3, 4, 5)
    y <- c(6, 7, 8, 9)
    x + y
                        My Guess: 8 10 12 14           Actual: 8 10 12 14
```

Exercise 16: Guess what the result of each of the following will be, then check your answers

```
a) x <- c(2, 3, 4, 5)
   x + 1
                        My Guess: 3 4 5 6              Actual: 3 4 5 6
b) x <- c(2, 3, 4, 5)
   x * 2
                        My Guess: 4 6 8 10             Actual: 4 6 8 10
```

Exercise 17: Guess what the result of the following will be, then check your answer

```
y <- c(6, 7, 8)
z <- c(2, 3)
y + z                   # My Guess: 8 10 10            Actual: 8 10 10
```

Exercise 18: In R, single-valued variables and constants are vectors of length one.
Guess what the result of each of the following will be, then check your answers

```
a) x <- 2
   is.vector(x)
                        My Guess: TRUE                 Actual: TRUE
b) is.vector(2)
                        My Guess: TRUE                 Actual: TRUE
```

Section 4.3

Exercise 19: Guess what the result of each of the following will be,
then check your answers

```
a)  x <- c(2, 3, "a")
    x
                        My Guess: "2" "3" "a"          Actual: "2" "3" "a"

b) x <- c(2, 3, TRUE)
   x
                        My Guess: 2 3 1                Actual: 2 3 1

c) x <- c("a", "b")
   y <- c(FALSE, TRUE)
   c(x, y)
                        My Guess: "a" "b" "FALSE" "TRUE"   Actual: "a" "b" "FALSE"
```

"TRUE"

Section 4.4
Exercise 20 Consider the vector: x <- c(7, 6, 4, 2, 3, 5)

```
a) x[2]                 My Guess: 6                     Actual: 6
b) x[-2]                My Guess: 7 4 2 3 5             Actual: 7 4 2 3 5
c) x[c(1, 2)]           My Guess: 7 6                   Actual: 7 6
d) x[c(2, 1)]           My Guess: 6 7                   Actual: 6 7
e) x[1] <- 5            My Guess: 5 6 4 2 3 5           Actual: 5 6 4 2 3 5
   x
```

Exercise 21: Consider the vector
x <- c(7, 6, 4, 2, 3, 5)
a) Write a command that returns the 4th element of x.

```
x[4]            # Output: 2
```

b) Write a command that replaces the 4th element of x with the value 1.

```
x[4] <- 1       # Output: 7 6 4 1 3 5
```

c) Write a command that returns all but the 6th element of x.

```
x[-6]           # Output: 7 6 4 1 3
```

Exercise 22: Consider the vector. Guess the output of the following commands.

```
x <- c(7, 6, 4, 2)
x[c(2, 1, 3, 4)]
                        My Guess: 6 7 4 2               Actual: 6 7 4 2
```

Exercise 23: Consider the vector
x <- c(7, 6, 4, 2, 3, 5)
a) Write a command using sort() that returns the elements of x sorted in ascending order.

```
sort(x)         # Output: 2, 3, 4, 5, 6, 7
```

b) Write a command using rev() that returns the elements of x in reverse order.

```
rev(x)          # Output: 5, 3, 2, 4, 6, 7
```

c) Look at the help page for sort() by typing ? sort Write a
command using sort() that returns the elements of x in descending order by setting decreasing
= TRUE.

```
sort(x, decreasing = TRUE) # Output: 7, 6, 5, 4, 3, 2
```

Exercise 24: Consider the vector
x <- c(7, 6, 4, 2, 3, 5)
Guess what the result of the following will be, then check your answer:

```
x[c(FALSE, FALSE, TRUE, FALSE, FALSE, TRUE)]
                        My Guess: 4 5                   Actual: 4 5
```

Exercise 25: Guess what the result of each of the following will be, then check your answers:

```
a) 1:5                  My Guess: 1 2 3 4 5             Actual: 1 2 3 4 5
b) 6:10                 My Guess: 6 7 8 9 10            Actual: 6 7 8 9 10
c) 5:1                  My Guess: 5 4 3 2 1             Actual: 5 4 3 2 1
```

Exercise 26: Guess what the result of the following will be, then check your answer:

```
is.vector(1:5)
                        My Guess: TRUE                  Actual: TRUE
```

Exercise 27: Guess what the result of each of the following will be, then check your answers:

```
a) seq(from = 1, to = 2.5, by = 0.5)
                        My Guess: 1 1.5 2 2.5           Actual: 1.0 1.5 2.0 2.5

b) seq(from = 2.5, to = 1, by = -0.5) # Note that by is negative
                        My Guess: 2.5 2.0 1.5 1.0       Actual: 2.5 2.0 1.5 1.0
```

Exercise 28: Guess what the result of each of the following will be, then check your answers:

```
a) rep(2, times = 3)    My Guess: 2 2 2                 Actual: 2 2 2
b) rep(1:2, times = 3)  My Guess: 1 2 1 2 1 2           Actual: 1 2 1 2 1 2
```

Section 4.5

Exercise 29: Consider the following vector:

```
x <- c(3, 4, 10)
```

Guess what the result of each of the following will be, then check your answers:

```
a) x == 4               My Guess: FALSE TRUE FALSE      Actual: FALSE TRUE FALSE
b) x > 4                My Guess: FALSE FALSE TRUE      Actual: FALSE FALSE TRUE
c) x >= 4               My Guess: FALSE TRUE TRUE       Actual: FALSE TRUE TRUE
d) x != 4               My Guess: TRUE FALSE TRUE       Actual: TRUE FALSE TRUE
```

Exercise 30: Consider the following two vectors:

```
x <- c(3, 4, 10)
y <- c(3, 4, 5)
```

Guess what the result of each of the following will be, then check your answers:

```
a) x == y               My Guess: TRUE TRUE FALSE       Actual: TRUE TRUE FALSE
b) x != y               My Guess: FALSE FALSE TRUE      Actual: FALSE FALSE TRUE
```

Exercise 31: Recall that R coerces TRUE and FALSE to 1 and 0, respectively, when it needs to. Guess
what the result of each of the following will be, then check your answers:

```
a) TRUE + TRUE + FALSE + FALSE + FALSE          My Guess: 2     Actual: 2
b) sum(c(TRUE, TRUE, FALSE, FALSE, FALSE))      My Guess: 2     Actual: 2
```

Exercise 32: Consider the following vector:

```
x <- c(10, 8, -2, -6, -5)
```

Guess what will be returned by of each of the following commands, then check your answers:

```
a) x > 0
My Guess: TRUE TRUE FALSE FALSE FALSE           Actual: TRUE  TRUE FALSE FALSE FALSE
```

```
b) sum(x > 0)           My Guess: 2             Actual: 2
```

Section 4.6

Exercise 33: Consider the vector:

```
x <- c(2, 8, 6, 7, 1, 4, 9)
```

Guess what each of the following commands will return, then check your answers:

```
a) any(x == 4)          My Guess: TRUE                  Actual: TRUE
b) all(x == 4)          My Guess: FALSE                 Actual: FALSE
c) which(x == 4)        My Guess: 6                     Actual: 6
d) which(x != 4)        My Guess: 1 2 3 4 5 7           Actual: 1 2 3 4 5 7
```

Exercise 34: Consider the vectors:

```
x <- c(53, 42, 64, 71, 84, 62, 95)
y <- c(53, 41, 68, 71, 81, 66, 65)
```

a) Write a command involving any() and == to determine if any of the values in x are equal to their
corresponding value in y.

```
any(x == y)             # Output: TRUE
```

b) Write a command involving all() and == to determine if all of the values in x are equal to their
corresponding value in y.

```
all(x == y)             # Output: FALSE
```

c) Write a command involving which() and == to determine which of the values in x are equal to their
corresponding value in y.

```
which(x == y)           # Output: 1 4
```

Exercise 35: Consider the vector:

```
x <- c(53, 42, 64, 71, 84, 62, 95)
```

Guess what the result of each of the following commands will be, then check your answers:

```
a) which.min(x)         My Guess: 2                     Actual: 2
b) which.max(x)         My Guess: 7                     Actual: 7
```

Section 4.7

Exercise 36: Consider the following data set:

```
x <- c(10, 147, 7, 6, 7, 12, 9, 12, 11, 8)
```

a) Use mean() to compute the mean.

```
mean(x)                 # Output: 22.9
```

b) Use median() to compute the median.

```
median(x)               # Output: 9.5
```

c) Use sd() to compute the standard deviation of x

```
sd(x)                   # Output: 43.65636
```

Exercise 37: The standard deviation measures variation in a set of data.

a) What do you think the standard deviation of the following data set will be? Check

your answer
using sd().

```
u <- c(5, 5, 5, 5, 5)           My Guess: 0             Actual: 0
```

b) Which of the following two data sets do you think will have a larger standard deviation?
Check your answer using sd().

```
u <- c(5, 6, 7)         # Output of sd(u): 1
v <- c(1, 6, 11)        # Output of sd(v): 5
```

Data set v will have the larger standard deviation.

Section 4.8
Exercise 38: The function abs() takes the absolute value of a number. It’s a vectorized function.
Guess what the result of the following command will be, then check your answer:

```
x <- c(-1, 3, -4, -2)
abs(x)
                                My Guess: 1 3 4 2       Actual: 1 3 4 2
```

Exercise 39: Consider the following temperature measurements, in degrees Celsius:

```
degreesC <- c(23, 19, 21, 22, 18, 20, 24, 25)
```

Describe in words what the following command will do to the Celsius temperatures? Try it.

```
degreesF <- (9/5) * degreesC + 32
```

The above command will store the results of the computation from the given vector degreesC into
degreesF. The computation will be done to every number in the vector degreesC.

Section 4.9
Exercise 40: Consider again the vector:

```
x <- c(538, 432, 684, 716, 814, 624, 956)
```

a) Guess values will be returned by the following command, then check your answer:

```
x[x > 700]                      My Guess: 716 814 956   Actual: 716 814 956
```

b) Guess values will be returned by the following command, then check your answer:

```
subset(x, subset = x > 700)     My Guess: 716 814 956   Actual: 716 814 956
```

c) Write a command involving square brackets [ ] that extracts from x all the values that
are not equal to 814. Hint: Use the the comparison operator !=.

```
x[x != x[5]] or x[x != 814]     # Output: 538 432 684 716 624 956
```

Exercise 41: Consider this data set, showing the genders, ages, and systolic blood pressures for 12 people:

Gender	Age	Blood Pressure
f	33	118
m	35	115
f	29	110
m	34	117
m	37	112

f	36	119
f	35	114
f	40	121
m	43	123
f	38	117
f	40	120
m	44	121

Here are the same data:

```
Gender <- c("f", "m", "f", "m", "m", "f", "f", "f", "m", "f", "f", "m")
Age <- c(33, 35, 29, 34, 37, 36, 35, 40, 43, 38, 40, 44)
BP <- c(118, 115, 110, 117, 112, 119, 114, 121, 123, 117, 120, 121)
```

Which does which?

```
Age[BP > 117]           # Extracts ages of people whose blood pressure is greater than 117
BP[Gender == "m"]       # Extracts Blood pressure of people who are males
```

Section 4.10

Exercise 42: Guess what the result of each of the following will be, then check your answers.

```
a) 3 == NA              My Guess: NA            Actual: NA
b) NA == NA             My Guess: NA            Actual: NA
```

Exercise 43: Consider the vector:

```
x <- c(1, 2, NA)
```

Guess what the result of each of the following will be, then check your answers.

```
a) is.na(x)             My Guess: FALSE FALSE TRUE      Actual: FALSE FALSE TRUE

b) x[is.na(x)] <- 0 # Note that is.na(x) is a "logical" vector.
   x                    My Guess: 1 2 0         Actual: 1 2 0
```

Exercise 44: Consider the following vector:

```
x <- c(1, 2, NA)
```

a) Guess what the result of the following command will be, then check your answer:

```
sum(x)                  My Guess: NA            Actual: NA
```

b) Guess what the result of the following command will be, then check your answer:

```
sum(x, na.rm = TRUE)
                        My Guess: 3             Actual: 3
```

Section 5.1

Exercise 45: Here’s a matrix x:

```
x <- matrix(c(8, 8, 8, 4, 4, 4), nrow = 2, byrow = TRUE)
x
##      [,1]    [,2]    [,3]
## [1,] 8       8       8
## [2,] 4       4       4
```

Guess what the result of each of the following will be, then check your answers.

```
a) dim(x)               My Guess: 2 3           Actual: 2 3
b) nrow(x)              My Guess: 2             Actual: 2
```

```
c) ncol(x)              My Guess: 3             Actual: 3
```

Exercise 46: What happens when you run the following command?

```
matrix(c(1, 2, 3), nrow = 4, ncol = 2)
Output:
     [,1] [,2]
[1,]    1    2
[2,]    2    3
[3,]    3    1
[4,]    1    2
```

Exercise 47: Here’s a matrix x:

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

Do you think the following commands will all produce the matrix above? Check your answer.

```
x <- matrix(c(5, 4, 5, 4), nrow = 2, ncol = 2)          # This produces the matrix
above
x <- cbind(c(5, 4), c(5, 4))                            # This produces the matrix
above
x <- rbind(c(5, 5), c(4, 4))                            # This produces the matrix
above
```

All three will create the matrix above.

Section 5.2

Exercise 48: Consider the following matrix:

```
x <- matrix(1:9, nrow = 3, ncol = 3)
x
##      [,1]    [,2]    [,3]
## [1,] 1       4       7
## [2,] 2       5       8
## [3,] 3       6       9
```

Guess what the result of each of the following will be, then check your answers.

```
a) x[1, 3]                      My Guess: 7             Actual: 7
b) x[1, ]                       My Guess: 1 4 7         Actual: 1 4 7
c) x[, 3]                       My Guess: 7 8 9         Actual: 7 8 9
d) x[, -3]
My guess:                       Actual:
        [,1]    [,2]                    [,1]    [,2]
[1,]    1       4               [1,]    1       4
[2,]    2       5               [2,]    2       5
[3,]    3       6               [3,]    3       6
```

Exercise 49: Consider the following matrix:

```
x <- matrix(1:6, nrow = 2, ncol = 3)
x
##      [,1]    [,2]    [,3]
## [1,] 1       3       5
## [2,] 2       4       6
```

What does the following command do? Check your answer.

```
x[, c(3, 1, 2)]
```

My Guess:

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

Actual:

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

Section 5.3

Exercise 50: Consider the following matrix x:

```
x <- matrix(c(8, 6, 3, 6, 5, 7), nrow = 3, ncol = 2)
x
##      [,1]    [,2]
## [1,] 8       6
## [2,] 6       5
## [3,] 3       7
```

Guess what the result of each of the following will be, then check your answers.

```
a) apply(x, MARGIN = 1, FUN = sum)              My Guess: 14 11 10
Actual: 14 11 10
b) apply(x, MARGIN = 2, FUN = min)              My Guess: 3 5
Actual: 3 5
```

Exercise 51:

a) Which of the following commands finds the mean expenditure for each of the five expenditure categories? Hint: Should MARGIN be 1 or 2?

```
apply(X = USPersonalExpenditure, MARGIN = 1, FUN = mean)        # This should be the one used
apply(X = USPersonalExpenditure, MARGIN = 2, FUN = mean)
```

MARGIN = 1 should be used to find the mean expenditure for each of the five expenditures.

b) Which of the following commands finds the total expenditure for each of the five years? Hint: Should MARGIN be 1 or 2?

```
apply(X = USPersonalExpenditure, MARGIN = 1, FUN = sum)
apply(X = USPersonalExpenditure, MARGIN = 2, FUN = sum)         # This should be the one used
```

MARGIN = 2 should be used to find the sum for each of the five years.

Section 6.1

Exercise 52:

```
Employees <- list(Name = c("Joe", "Kim", "Ann", "Bob"),
                  Salary = c(56000, 67000, 60000, 55000),
                  Union = c (TRUE, TRUE, FALSE, FALSE))
```

a) Use str() to look at the structure of the list. Report the results.

Results:

```
List of 3
 $ Name  : chr [1:4] "Joe" "Kim" "Ann" "Bob"
 $ Salary: num [1:4] 56000 67000 60000 55000
 $ Union : logi [1:4] TRUE TRUE FALSE FALSE
```

b) Use length() to find the number of elements in the list. Report the result.

Results:

```
[1] 3
```

Section 6.2

Exercise 53: Recreate the Employees list from Exercise 52.

a) Guess what the result of the following command will be, then check your answer:

```
Employees[[2]]
My Guess: 56000 67000 60000 55000       Actual: 56000 67000 60000 55000
```

b) Guess what the result of the following command will be, then check your answer:

```
Employees$Salary
My Guess: 56000 67000 60000 55000       Actual: 56000 67000 60000 55000
```

c) Write a command involving [[ ]] that returns the "logical" vector Union from the Employees list.

```
Employees[[3]]          # Output: TRUE  TRUE FALSE FALSE
```

d) Now write a command involving $ that returns the "logical" vector Union from the Employees list.

```
Employees$Union         # Output: TRUE  TRUE FALSE FALSE
```

Section 6.3

Exercise 54: Here’s a simple list x:

```
x <- list(x1 = c(1, 2), x2 = c("a", "b"), x3 = 12)
```

What will the following command return? Check your answer.

```
names(x)                         My Guess: "x1" "x2" "x3"                Actual: "x1"
"x2" "x3"
```

Section 6.4

Exercise 55: Here’s the HtWtAge list again:

```
HtWtAge <- list(Height = c(65, 68, 70, 60, 61),
                Weight = c(160, 171, 158, 148, 215),
                Age = c(23, 20, 37, 40, 44))
```

a) Write a command using lapply(), with FUN = max, that returns a list containing the maximum value of each variable (Height, Weight, and Age).

```
lapply(X = HtWtAge, FUN = max)
# Output:
# $Height
# [1] 70
#
```

```
# $Weight
# [1] 215
#
# $Age
# [1] 44
```

b) Now write a command using sapply() that does the same thing, but returns a vector

```
 sapply(X = HtWtAge, FUN = max)
# Output:
# Height Weight    Age
#    70    215     44
```

Section 7.1
Exercise 56:
a) Here’s the mice data set as three vectors:

```
col <- c("white", "grey", "black", "brown", "black", "white", "black", "white")
wt <- c(23, 21, 12, 26, 25, 22, 26, 19)
len <- c(3.8, 3.7, 3.0, 3.4, 3.4, 3.1, 3.5, 3.2)
```

Write a command using data.frame().Check that you created the data frame correctly by typing its name on the command line, for example:

```
mice.data <- data.frame(col, wt, len)           # Code to produce the data frame
mice.data

# Output:
#     col wt len
# 1 white 23 3.8
# 2  grey 21 3.7
# 3 black 12 3.0
# 4 brown 26 3.4
# 5 black 25 3.4
# 6 white 22 3.1
# 7 black 26 3.5
# 8 white 19 3.2
```

b) The file mice.txt contains the mice data set. After saving the file onto your computer, type the following:

```
my.file <- file.choose()
```

then in the dialog box choose the mice.txt file that you saved. Now type: my.file and report the results.
Results:

```
[1] "C:\\Users\\tboggess\\Downloads\\mice.txt"
```

c) Now try the following:

```
mice.data <- read.csv(my.file, sep = "", header = TRUE)
```

Make sure to check that the data were read in correctly, for example by typing:

```
mice.data
```

Results:

```
  Color Weight Length
1 white     23    3.8
2  grey     21    3.7
3 black     18    3.0
4 brown     26    3.4
5 black     25    3.4
6 white     22    3.1
7 black     26    3.5
8 white     19    3.2
```

d) Now type the following commands and report the results:

Results:

```
> nrow(mice.data) # Number of rows.
[1] 8
> ncol(mice.data) # Number of columns.
[1] 3
> head(mice.data) # First six rows.
  Color Weight Length
1 white     23    3.8
2  grey     21    3.7
3 black     18    3.0
4 brown     26    3.4
5 black     25    3.4
6 white     22    3.1
> names(mice.data) # Column names.
[1] "Color"  "Weight" "Length"
> str(mice.data) # "Structure".
'data.frame':   8 obs. of  3 variables:
 $ Color : chr  "white" "grey" "black" "brown" ...
 $ Weight: int  23 21 18 26 25 22 26 19
 $ Length: num  3.8 3.7 3 3.4 3.4 3.1 3.5 3.2
```

Section 7.2

Exercise 57: Consider the following data on nine people

Status	Age	Education
Married	36	HS Diploma
Single	33	Bachelor of Arts
Single	21	Bachelor of Science
Married	29	Bachelor of Science
Single	19	HS Diploma
Married	35	Bachelor of Arts

Married	39	Master of Science
Single	28	HS Diploma
Single	21	HS Diploma

The following commands will create a data frame containing the data:

```
status <- c("Married", "Single", "Single", "Married", "Single",
                        "Married", "Married", "Single", "Single")
age <- c(36, 33, 21, 29, 19, 35, 39, 28, 21)
educ <- c("HS Diploma", "Bachelor of Arts", "Bachelor of Science",
        "Bachelor of Science", "HS Diploma", "Bachelor of Arts",
        "Master of Science", "HS Diploma", "HS Diploma")
my.data <- data.frame(Status = status, Age = age, Education = educ)
```

a) Guess what each of the following commands will return, then check your answers:

```
my.data[6, 2]
My Guess: 35            Actual: 35

my.data[6, ]
          Status  Age   Education               Status  Age     Education

My Guess: Married 35 Bachelor of Arts   Actual: Married 35 Bachelor of Arts

my.data[, 2]
My Guess: 36 33 21 29 19 35 39 28 21    Actual: 36 33 21 29 19 35 39 28 21

my.data$Age
My Guess: 36 33 21 29 19 35 39 28 21    Actual: 36 33 21 29 19 35 39 28 21

my.data[[2]]
My Guess: 36 33 21 29 19 35 39 28 21    Actual: 36 33 21 29 19 35 39 28 21
```

b) Write three different commands that return the entire 3rd column (Education) of the data frame:

Using single square brackets [ ].

```
my.data[, 3]
# Output:
[1] "HS Diploma"          "Bachelor of Arts"
[3] "Bachelor of Science" "Bachelor of Science"
[5] "HS Diploma"          "Bachelor of Arts"
[7] "Master of Science"   "HS Diploma"
[9] "HS Diploma"
```

Using the dollar sign operator $.

```
my.data$Education
# Output:
[1] "HS Diploma"          "Bachelor of Arts"
[3] "Bachelor of Science" "Bachelor of Science"
[5] "HS Diploma"          "Bachelor of Arts"
[7] "Master of Science"   "HS Diploma"
```

```
[9] "HS Diploma"
```

Using double square brackets [[ ]].

```
my.data[[3]]
# Output:
[1] "HS Diploma"          "Bachelor of Arts"
[3] "Bachelor of Science" "Bachelor of Science"
[5] "HS Diploma"          "Bachelor of Arts"
[7] "Master of Science"   "HS Diploma"
[9] "HS Diploma"
```

c) The nine people have each aged one year since the data were collected. Here’s a vector containing
their current ages:

```
age2 <- c(37, 34, 22, 30, 20, 36, 40, 29, 22)
```

Describe in words what the following commands do.

```
my.data$Age <- NULL
```

The my.data$Age <- NULL command will delete the Age column from the data frame

```
my.data$Age2 <- age2
```

The my.data$Age2 <- age2 command will add age2 to the end of the data frame

d) Here’s another vector:

```
ageofspouse <- c(39, NA, NA, 34, NA, 27, 30, NA, NA)
```

Describe in words what the following commands both do.

```
my.data$AgeOfSpouse <- ageofspouse
```

The ageofspouse vector gets concatenated to the end of the data frame of my.data.

```
my.data[["AgeOfSpouse"]] <- ageofspouse
```

The ageofspouse vector gets concatenated to the end of the data frame of my.data.

Section 7.3
Exercise 58: Create the following data frame:

```
x <- data.frame(A = 1:5, B = 6:10, C = c("a", "b", "c", "d", "e"))
```

a) Guess what each of the following commands will return, then check your answers:

```
names(x)                        My Guess: "A" "B" "C"           Actual: "A" "B" "C"
is.vector(names(x))             My Guess: TRUE                  Actual: TRUE
typeof(names(x))                My Guess: "character"           Actual: "character"
```

b) Guess what the following will do, then check your answer:

```
names(x) <- c("AA", "BB", "CC")        My Guess: "AA" "BB" "CC"        Actual: "AA"
"BB" "CC"
names(x)
```

Section 7.5
Exercise 59:

```
breaks          The number of breaks per loom, where a loom corresponds to
```

	a fixed length of yarn.
wool	The type of wool (A or B)
tension	The level of tension (L, M, H)

a) Write a command involving square brackets [ ] that returns just the rows of warpbreaks
corresponding to observations made at the "M" level of tension.

```
warpbreaks[warpbreaks$tension == "M", ]
# Output:
   breaks wool tension
10     18    A       M
11     21    A       M
12     29    A       M
13     17    A       M
14     12    A       M
15     18    A       M
16     35    A       M
17     30    A       M
18     36    A       M
37     42    B       M
38     26    B       M
39     19    B       M
40     16    B       M
41     39    B       M
42     28    B       M
43     21    B       M
44     39    B       M
45     29    B       M
```

b) Now write a command that uses subset() to do the same thing as in part a.

```
subset(warpbreaks, subset = tension == "M")
# Output:
   breaks wool tension
10     18    A       M
11     21    A       M
12     29    A       M
13     17    A       M
14     12    A       M
15     18    A       M
16     35    A       M
17     30    A       M
18     36    A       M
37     42    B       M
38     26    B       M
39     19    B       M
40     16    B       M
41     39    B       M
42     28    B       M
```

```
43     21    B       M
44     39    B       M
45     29    B       M
```

Section 8.1

Exercise 60:

a) Write a command that uses factor() to convert the following "character" vector x.char <- c("a", "a", "b", "b", "c", "c", "d", "d") to a factor named x.factor:

```
x.factor
## [1] a a b b c c d d
## Levels: a b c d
x.factor <- factor(x.char)              # Command to convert character vector to
factor()
```

b) After creating the factor x.factor, guess what the result of the following command will be, then
check your answer:

```
length(x.factor)                # My Guess: 8           Actual: 8
```

c) Guess what the result of the following command will be, then check your answer:

```
levels(x.factor)                # My Guess: "a" "b" "c" "d"     Actual: "a" "b" "c"
"d"
```

d) Guess what the result of the following command will be, then check your answer:

```
nlevels(x.factor)               # My Guess: 4           Actual: 4
```

Exercise 61: Guess what the result of the following commands will be, then check your answer:

```
x.fac <- factor(c("a", "a", "a", "b", "b", "c"))
levels(x.fac)                   # My Guess: "a" "b" "c"  Actual: "a" "b" "c"
```

Exercise 62:

```
illit <- state.x77[ , 3]
murder <- state.x77[ , 5]
```

Report your R command.

```
 plot(x = illit, y = murder,                    # command to plot Murder rate Vs.
Illiteracy Rate
+ main = "Murder Rate Versus Illiteracy Rates",
+ xlab = "Illiteracy Rates",
+ ylab = "Murder Rates",
+ xlim = c(0, 4),
+ ylim = c(0, 17))
```

Exercise 63:

```
laughed <- c("Yes", "Yes", "Yes", "No", "Yes", "No", "Yes", "Yes", "Yes", "Yes",
"No",
"Yes", "Yes", "Yes", "No", "No", "Yes", "Yes", "Yes", "No", "Yes")
```

a) In words, what do the following commands do:

```
my.tab <- table(laughed)
my.tab
```

The commands above count the number of no's and yes's taken from the poll of people who answered yes or no to laughing.

b) Pass the table object my.tab to barplot() to make a bar plot of the counts. Report your R command.

```
barplot(my.tab)         # Displays amount of people who said yes or no in barchart
```

c) Pass the table object my.tab to pie() to make a pie chart of the counts. Report your R command.

```
pie(my.tab)             # Displays amount of people who said yes or no in pie chart
```

Section 10.0

Exercise 64: In R, everything that ”exists” is an object, and each object belongs to a class of objects. Guess the class of each of the objects below, then check your answers.

```
a) u <- c("a", "b", "c")
class(u)                                My Guess: "character"           Actual:
"character"

b) x <- c(3, 6, 1)
class(x)                                My Guess: "numeric"             Actual:
"numeric"

c) y <- list(3, 5, 2)
class(y)                                My Guess: "list"                Actual:
"list"
```

Exercise 65: In R, everything that ”exists” is an object, and each object belongs to a class of objects. Even functions are objects.

a) Guess the class of sqrt(), then check your answer.

```
class(sqrt)                             My Guess: "function"            Actual:
"function"
```

b) Guess what will be returned by the following command, then check your answer.

```
is.function(sqrt)                       My Guess: TRUE                  Actual: TRUE
```

c)

```
apply.fun <- function(x, FUN) {
        do.call(FUN, args = list(x))
}
```

After creating apply.fun(), guess what the result of each of the following commands

will be, then
check your answers:

```
apply.fun(x = 4, FUN = sqrt)            My Guess: 2                     Actual: 2
apply.fun(x = -3, FUN = abs)            My Guess: 3                     Actual: 3
```

Exercise 66:

```
x <- c("a", "b", "c")
y <- factor(c("a", "b", "c"))
```

a) Guess what the result of each of the following commands will be, then check your answers:

```
class(x)                                My Guess: "character"           Actual:
"character"
typeof(x)                               My Guess: "character"           Actual:
"character"
```

b) Guess what the result of each of the following commands will be, then check your answers:

```
class(y)                                My Guess: "factor"              Actual:
"factor"
typeof(y)                               My Guess: "integer"             Actual:
"integer"
```

Section 11.2

Exercise 67: The function summary() is a generic function, meaning it does something
different depending on the class of the object passed to it.

a) Use methods() to look at summary()’s methods. Is there a method for "data.frames"? How about
for "factors"?

```
# Output:
# Output for part a:
[1] summary.aov
 [2] summary.aovlist*
 [3] summary.aspell*
 [4] summary.check_packages_in_dir*
 [5] summary.connection
 [6] summary.data.frame
 [7] summary.Date
 [8] summary.default
 [9] summary.ecdf*
[10] summary.factor
[11] summary.glm
[12] summary.infl*
[13] summary.lm
[14] summary.loess*
[15] summary.manova
[16] summary.matrix
```

```
[17] summary.mlm*
[18] summary.nls*
[19] summary.packageStatus*
[20] summary.POSIXct
[21] summary.POSIXlt
[22] summary.ppr*
[23] summary.prcomp*
[24] summary.princomp*
[25] summary.proc_time
[26] summary.srcfile
[27] summary.srcref
[28] summary.stepfun
[29] summary.stl*
[30] summary.table
[31] summary.tukeysmooth*
[32] summary.warnings
```

Both summary.data.frame and summary.factor is listed as part of the methods included

in methods(summary).

b) Try the following commands, and describe the results:

```
x <- data.frame(x1 = c(4, 3, 6),
                x2 = c(4, 7, 9),
                x3 = c(1, 1, 3))
summary(x)
y <- factor(c("a", "a", "b", "c", "c", "c", "c"))
summary(y)

# Output for summary(x):
        x1                      x2
 Min.   :3.000          Min.   :4.000
 1st Qu.:3.500          1st Qu.:5.500
 Median :4.000          Median :7.000
 Mean   :4.333          Mean   :6.667
 3rd Qu.:5.000          3rd Qu.:8.000
 Max.   :6.000          Max.   :9.000
        x3
 Min.   :1.000
 1st Qu.:1.000
 Median :1.000
 Mean   :1.667
 3rd Qu.:2.000
 Max.   :3.000

# Output for summary(y):
a b c
2 1 4
```

Summary(x) describes x1, x2, and x3 by displaying the minimum, 1st quarter, median, mean, 3rd quarter, and maximum from the data put into the data frame. Summary(y) displays the frequency of the different characters displayed.

c) Compare the results with those of part b.

```
summary.data.frame(x)
summary.factor(y)
```

The results are similar to part b) and shows the exact same information but it seemed like it took less time to complete.

Section 13.1

Exercise 68: Here are two variables x and y:

```
x <- 4
y <- 7
```

Guess what the result of each of the following will be, then check your answers.

```
a) x > 2 & y == 7                        My Guess: TRUE         Actual: TRUE
b) x < 0 | y == 7                        My Guess: TRUE         Actual: TRUE
c) !(x < 0)                              My Guess: TRUE         Actual: TRUE
```

Exercise 69: Guess what the result of each of thefollowing commands will be, then check your answers.

```
a) 10 < 20 | 15 < 16 & 9 == 10           My Guess: TRUE         Actual: TRUE
b) (10 < 20 | 15 < 16) & 9 == 10         My Guess: FALSE        Actual: FALSE
c) 4 < 3 & (5 < 6 | 8 < 9)               My Guess: FALSE        Actual: FALSE
d) (4 < 3 & 5 < 6) | 8 < 9               My Guess: TRUE         Actual: TRUE
```

Exercise 70: Recall that the logical operators operate elementwise on vectors. Guess what the result of each of the following will be, then check your answers.

```
a) c(FALSE, TRUE, FALSE) & c(TRUE, TRUE, FALSE)
My Guess: FALSE TRUE FALSE      Actual: FALSE TRUE FALSE

b) c(FALSE, TRUE, FALSE) | c(TRUE, TRUE, FALSE)
My Guess: TRUE TRUE FALSE       Actual: TRUE TRUE FALSE

c) !c(FALSE, TRUE, FALSE)
My Guess: TRUE FALSE TRUE       Actual: TRUE FALSE TRUE
```

Exercise 71: Here’s a vector x:

```
x <- c(1, 2, NA, 6, 3, NA, 5)
```

Describe in words what the following command does:

```
!is.na(x)

# Output:
TRUE  TRUE FALSE  TRUE  TRUE FALSE  TRUE
```

It will show TRUE or FALSE depending on if the value is NA or not in the vector where TRUE is that the vector has a number in that index and FALSE being there is a NA value in a particular index.

Exercise 72: Here’s a vector x:

```
x <- c(1, 2, 6, 5)
```

Guess what the result of each of the following commands will be, then check your answers.

```
a) !(x > 4)
My Guess: TRUE TRUE FALSE FALSE         Actual: TRUE TRUE FALSE FALSE

b) x > 4 & x < 6
My Guess: FALSE FALSE FALSE TRUE        Actual: FALSE FALSE FALSE TRUE

c) !(x > 4 & x < 6)
My Guess: TRUE TRUE TRUE FALSE          Actual: TRUE TRUE TRUE FALSE

d) x > 4 | x < 6
My Guess: TRUE TRUE TRUE TRUE           Actual: TRUE TRUE TRUE TRUE
```

Exercise 73: Here are two vectors, Gender and Age:

```
Gender <- c("m", "f", "m", "f", "f")
Age <- c(27, 34, 55, 21, 43)
```

Guess what each of the following commands will return, then check your answers.

```
a) Gender == "m" & Age > 40
My Guess: FALSE FALSE TRUE FALSE FALSE          Actual: FALSE FALSE TRUE FALSE FALSE

b) Gender == "m" | Age > 40
My Guess: TRUE FALSE TRUE FALSE TRUE            Actual: TRUE FALSE TRUE FALSE TRUE
```

Section 13.2

Exercise 74: Here’s a function:

```
f1 <- function(x) {
        y <- x + 1
        return(y)
}
```

If we replace return(y) by just y:

```
f2 <- function(x) {
        y <- x + 1
        y
}
```

does the function do the same thing?

The functions do the same thing and will output the same result.

Exercise 75: In the code below, which argument (x or z) is the formal argument and which is the actual argument?

```
g <- function(x) {
x^2 - 1
}
```

```
z <- 2
g(x = z) # Which argument, x or z, is formal and which is actual?
## [1] 3
```

The argument z is the actual and x is the formal argument.

Exercise 76:

a) Write a function that takes two arguments, x and y, and returns their relative difference,
defined as f(x, y), where | · | is the absolute value. Test your functions by passing it a
few different values for x and y.

```
# Returns relative difference between two numbers
f <- function(x, y) {
  result <- abs((x - y) / y)
  result # could be return(result)
}

# Tested Values:
x <- 3
y <- 2
f(x, y)

# Output:
[1] 0.5

x <- -3
y <- -4
f(x, y)

# Output:
[1] 0.25

x <- -2
y <- 3
f(x, y)

# Output:
[1] 1.666667
```

b) What happens when you pass it the value y = 0? What about when you pass it x = 0 and y = 0.
It displays inf when y = 0 and it displays NaN when x = 0 and y = 0.

c) Rewrite your function so that it specifies a default value of 1 for y.

```
# Returns relative difference between two numbers. If y is not specified, y is assumed to be 1.
f <- function(x, y = 1) {
  result <- abs((x - y) / y)
  result
}
```

Exercise 77: Write a function that takes a vector argument x and returns a list containing the mean,median, standard deviation, and range of x. Use mean(), median(), sd(), and range().

```
# Outputs mean, median, sd, and range from a vector
vectInfo <- function(x) {
  vec.mean <- mean(x)
  vec.median <- median(x)
  vec.sd <- sd(x)
  vec.range <- range(x)
  listOfX <- list(vec.mean, vec.median, vec.sd, vec.range)
  return(listOfX)
}
```

Exercise 78: Look at the help page for list() by typing ? list
What do the ...'s mean when it says list(...) under Usage?

The ... represents objects that may or may not be named and list(...) displays the objects in list form.

Exercise 79:
a) Write a function that takes two vectors x and y and returns the maximum value in the two vectors combined. Hint: Use max() with c(x, y).

```
# Takes in two vectors and outputs the largest number of the two vectors
maxVal <- function(x, y) {
  max.val <- max(c(x, y))
  max.val
}
```

b) Modify your function so that it uses ... to take a variable number of vectors as arguments, and returns the maximum value in all the vectors combined. Hint: Use max() with c(...).

```
# Takes in at least two vectors and outputs the largest number of the vectors
maxVal <- function(x, y, ...) {
  max.val <- max(c(x, y, ...))
  max.val
}
```