FDA Lab-7

KHAN MOHD OWAIS RAZA
20BCD7138

Q.1] Consider a numeric vector x <- c(3,4,5,6,7,8)
• Write a command to recode the values less than 6 with zero in the vector x.
• Write a command to recode the values between 4 and 8 with 100.
• Write a command to recode the values that are less than 5 or greater than 6 with 50 .
• Write a command to recode the values less than 6 with NA in the vector x.
• Write a command to recode the values between 4 and 8 with NA.
• Write a command to recode the values that are less than 5 or greater than 6 with NA.
• Count number of NA values after each operation .
• Find mean of x (Hint: exclude NA values).
• Find median of x (Hint: exclude NA values).
• Write a command to recode the values less than 6 with "NA" (enclose NA with double quotes) in the vector x.
• Write a command to recode the values between 4 and 8 with "NA" .
• Write a command to recode the values that are less than 5 or greater than 6 with "NA"
• Count number of NA values after each operation.
• Find mean of x (Hint: exclude NA values).
• Find median of x (Hint: exclude NA values).
• What is the difference between NA and "NA"?

```
# KHAN MOHD OWAIS RAZA
# 20BCD7138
# Numeric vector
x <- c(3, 4, 5, 6, 7, 8)
x
# (1) Recode values less than 6 with zero
x_recoded <- ifelse(x < 6, 0, x)
# Count number of NA values after recoding
num_na_1 <- sum(is.na(x_recoded))
x_recoded
```

```
# (2) Recode values between 4 and 8 with 100
x_recoded <- ifelse(x >= 4 & x <= 8, 100, x)
# Count number of NA values after recoding
num_na_2 <- sum(is.na(x_recoded))
x_recoded

# (3) Recode values less than 5 or greater than 6 with 50
x_recoded <- ifelse(x < 5 | x > 6, 50, x)
# Count number of NA values after recoding
num_na_3 <- sum(is.na(x_recoded))
x_recoded

# (4) Recode values less than 6 with NA
x_recoded <- ifelse(x < 6, NA, x)
# Count number of NA values after recoding
num_na_4 <- sum(is.na(x_recoded))
x_recoded

# (5) Recode values between 4 and 8 with NA
x_recoded <- ifelse(x >= 4 & x <= 8, NA, x)
# Count number of NA values after recoding
num_na_5 <- sum(is.na(x_recoded))
x_recoded

# (6) Recode values less than 5 or greater than 6 with NA
x_recoded <- ifelse(x < 5 | x > 6, NA, x)
# Count number of NA values after recoding
num_na_6 <- sum(is.na(x_recoded))
x_recoded

# (7) Count number of NA values after each operation
num_na_values <- c(num_na_1, num_na_2, num_na_3, num_na_4,
num_na_5, num_na_6)
num_na_values

# (8) Find mean of x (exclude NA values)
```

```r
mean_x <- mean(x, na.rm = TRUE)
mean_x

# (9) Find median of x (exclude NA values)
median_x <- median(x, na.rm = TRUE)
median_x

# (10) Recode values less than 6 with "NA" (enclose NA with
double quotes)
x_recoded <- ifelse(x < 6, "NA", x)
# Count number of NA values after recoding
num_na_10 <- sum(is.na(x_recoded))
x_recoded

# (11) Recode values between 4 and 8 with "NA"
x_recoded <- ifelse(x >= 4 & x <= 8, "NA", x)
# Count number of NA values after recoding
num_na_11 <- sum(is.na(x_recoded))
x_recoded

# (12) Recode values less than 5 or greater than 6 with "NA"
x_recoded <- ifelse(x < 5 | x > 6, "NA", x)
# Count number of NA values after recoding
num_na_12 <- sum(is.na(x_recoded))
x_recoded

# (13) Count number of NA values after each operation
num_na_values_2 <- c(num_na_10, num_na_11, num_na_12)
num_na_values_2

# (14) Find mean of x (exclude NA values)
mean_x_2 <- mean(x, na.rm = TRUE)
mean_x_2

# (15) Find median of x (exclude NA values)
median_x_2 <- median(x, na.rm = TRUE)
median_x_2
```

```
# (16) NA represents a missing value in R, while "NA" is a
character string.
# NA is used in R to denote missing or undefined values, while
# "NA" is simply a character representation of the string "NA".

> # KHAN MOHD OWAIS RAZA
> # 20BCD7138
> # Numeric vector
> x <- c(3, 4, 5, 6, 7, 8)
> x
[1] 3 4 5 6 7 8
> # (1) Recode values less than 6 with zero
> x_recoded <- ifelse(x < 6, 0, x)
> # Count number of NA values after recoding
> num_na_1 <- sum(is.na(x_recoded))
> x_recoded
[1] 0 0 0 6 7 8
>
> # (2) Recode values between 4 and 8 with 100
> x_recoded <- ifelse(x >= 4 & x <= 8, 100, x)
> # Count number of NA values after recoding
> num_na_2 <- sum(is.na(x_recoded))
> x_recoded
[1]   3 100 100 100 100 100
>
> # (3) Recode values less than 5 or greater than 6 with 50
> x_recoded <- ifelse(x < 5 | x > 6, 50, x)
> # Count number of NA values after recoding
> num_na_3 <- sum(is.na(x_recoded))
> x_recoded
[1] 50 50  5  6 50 50
>
> # (4) Recode values less than 6 with NA
> x_recoded <- ifelse(x < 6, NA, x)
> # Count number of NA values after recoding
> num_na_4 <- sum(is.na(x_recoded))
> x_recoded
```

```
[1] NA NA NA 6 7 8
>
> # (5) Recode values between 4 and 8 with NA
> x_recoded <- ifelse(x >= 4 & x <= 8, NA, x)
> # Count number of NA values after recoding
> num_na_5 <- sum(is.na(x_recoded))
> x_recoded
[1] 3 NA NA NA NA NA
>
> # (6) Recode values less than 5 or greater than 6 with NA
> x_recoded <- ifelse(x < 5 | x > 6, NA, x)
> # Count number of NA values after recoding
> num_na_6 <- sum(is.na(x_recoded))
> x_recoded
[1] NA NA 5 6 NA NA
>
> # (7) Count number of NA values after each operation
> num_na_values <- c(num_na_1, num_na_2, num_na_3, num_na_4,
num_na_5, num_na_6)
> num_na_values
[1] 0 0 0 3 5 4
>
> # (8) Find mean of x (exclude NA values)
> mean_x <- mean(x, na.rm = TRUE)
> mean_x
[1] 5.5
>
> # (9) Find median of x (exclude NA values)
> median_x <- median(x, na.rm = TRUE)
> median_x
[1] 5.5
>
> # (10) Recode values less than 6 with "NA" (enclose NA
with double quotes)
> x_recoded <- ifelse(x < 6, "NA", x)
> # Count number of NA values after recoding
> num_na_10 <- sum(is.na(x_recoded))
> x_recoded
```

```
[1] "NA" "NA" "NA" "6" "7" "8"
>
> # (11) Recode values between 4 and 8 with "NA"
> x_recoded <- ifelse(x >= 4 & x <= 8, "NA", x)
> # Count number of NA values after recoding
> num_na_11 <- sum(is.na(x_recoded))
> x_recoded
[1] "3" "NA" "NA" "NA" "NA" "NA"
>
> # (12) Recode values less than 5 or greater than 6 with
"NA"
> x_recoded <- ifelse(x < 5 | x > 6, "NA", x)
> # Count number of NA values after recoding
> num_na_12 <- sum(is.na(x_recoded))
> x_recoded
[1] "NA" "NA" "5" "6" "NA" "NA"
>
> # (13) Count number of NA values after each operation
> num_na_values_2 <- c(num_na_10, num_na_11, num_na_12)
> num_na_values_2
[1] 0 0 0
>
> # (14) Find mean of x (exclude NA values)
> mean_x_2 <- mean(x, na.rm = TRUE)
> mean_x_2
[1] 5.5
>
> # (15) Find median of x (exclude NA values)
> median_x_2 <- median(x, na.rm = TRUE)
> median_x_2
[1] 5.5
>
> # (16) NA represents a missing value in R, while "NA" is a
character string.
> # NA is used in R to denote missing or undefined values,
while
> # "NA" is simply a character representation of the string
"NA".
```

R Console

```
> # KHAN MOHD OWAIS RAZA
> # 20BCD7138
> # Numeric vector
> x <- c(3, 4, 5, 6, 7, 8)
> x
[1] 3 4 5 6 7 8
> # (1) Recode values less than 6 with zero
> x_recoded <- ifelse(x < 6, 0, x)
> # Count number of NA values after recoding
> num_na_1 <- sum(is.na(x_recoded))
> x_recoded
[1] 0 0 0 6 7 8
>
> # (2) Recode values between 4 and 8 with 100
> x_recoded <- ifelse(x >= 4 & x <= 8, 100, x)
> # Count number of NA values after recoding
> num_na_2 <- sum(is.na(x_recoded))
> x_recoded
[1]   3 100 100 100 100 100
>
> # (3) Recode values less than 5 or greater than 6 with 50
> x_recoded <- ifelse(x < 5 | x > 6, 50, x)
> # Count number of NA values after recoding
> num_na_3 <- sum(is.na(x_recoded))
> x_recoded
[1] 50 50  5  6 50 50
>
> # (4) Recode values less than 6 with NA
> x_recoded <- ifelse(x < 6, NA, x)

> # Count number of NA values after recoding
> num_na_4 <- sum(is.na(x_recoded))
> x_recoded
[1] NA NA NA  6  7  8
>
> # (5) Recode values between 4 and 8 with NA
> x_recoded <- ifelse(x >= 4 & x <= 8, NA, x)
> # Count number of NA values after recoding
> num_na_5 <- sum(is.na(x_recoded))
> x_recoded
[1]  3 NA NA NA NA NA
>
> # (6) Recode values less than 5 or greater than 6 with NA
> x_recoded <- ifelse(x < 5 | x > 6, NA, x)
> # Count number of NA values after recoding
> num_na_6 <- sum(is.na(x_recoded))
> x_recoded
[1] NA NA  5  6 NA NA
>
> # (7) Count number of NA values after each operation
> num_na_values <- c(num_na_1, num_na_2, num_na_3, num_na_4, num_na_5, num_na_6)
> num_na_values
[1] 0 0 0 3 5 4
>
> # (8) Find mean of x (exclude NA values)
> mean_x <- mean(x, na.rm = TRUE)
> mean_x
[1] 5.5
```

```
>
> # (9) Find median of x (exclude NA values)
> median_x <- median(x, na.rm = TRUE)
> median_x
[1] 5.5
>
> # (10) Recode values less than 6 with "NA" (enclose NA with double quotes)
> x_recoded <- ifelse(x < 6, "NA", x)
> # Count number of NA values after recoding
> num_na_10 <- sum(is.na(x_recoded))
> x_recoded
[1] "NA" "NA" "NA" "6"  "7"  "8"
>
> # (11) Recode values between 4 and 8 with "NA"
> x_recoded <- ifelse(x >= 4 & x <= 8, "NA", x)
> # Count number of NA values after recoding
> num_na_11 <- sum(is.na(x_recoded))
> x_recoded
[1] "3"  "NA" "NA" "NA" "NA" "NA"
>
> # (12) Recode values less than 5 or greater than 6 with "NA"
> x_recoded <- ifelse(x < 5 | x > 6, "NA", x)
> # Count number of NA values after recoding
> num_na_12 <- sum(is.na(x_recoded))
> x_recoded
[1] "NA" "NA" "5"  "6"  "NA" "NA"
>
> # (13) Count number of NA values after each operation
> num_na_values_2 <- c(num_na_10, num_na_11, num_na_12)
> num_na_values_2
[1] 0 0 0
>
> # (14) Find mean of x (exclude NA values)
> mean_x_2 <- mean(x, na.rm = TRUE)
> mean_x_2
[1] 5.5
>
> # (15) Find median of x (exclude NA values)
> median_x_2 <- median(x, na.rm = TRUE)
> median_x_2
[1] 5.5
>
> # (16) NA represents a missing value in R, while "NA" is a character string.
> # NA is used in R to denote missing or undefined values, while
> # "NA" is simply a character representation of the string "NA".
>
```

Q.2] Consider the dataset airquality

[1] Print the dataset airquality

[2] Print the structure of the dataset airquality

[3] Print the summary of all the variables of the dataset airquality (Hint: Use function summary())

[4] How many of the variables (columns) are in the dataset airquality ?

[5] How many observations (rows) are in the dataset airquality?

[6] What are the values getting displayed when we use summary() function?

[7] What is quartile & how to find them ?

[8] What are 1st and 3rd quartiles ?

[9] Copy the dataset airquality to aq (Better work on a copy of original data instead of working on original data to avoid the loss of information)

[10] Print the dataset aq

[11] Print the structure of the dataset aq

[12] Print the summary of all the variables of the dataset aq (Hint: Use function summary())

[13] Print top 6 observations

[14] Print last 6 observations

[15] Replace the NA values in the attribute Ozone in aq by zero

[16] Print the summary of all the variables of the dataset aq

[17] Replace the NA values in the attribute Ozone in aq by mean of the remaining values

[18] Print the summary of the dataset aq

[19] Copy the dataset airquality to aq1. Replace the NA values in the attribute Ozone in aq1 by median of the remaining values. Print the summary of the dataset aq1

[20] Copy the dataset airquality to aq2. Replace the NA values in the attribute Ozone in aq2 by mode of the remaining values. Print the summary of the dataset aq2

[21] Repeat the above five operations for the attribute Solar.R

[22] Replace all the values of Temp with global constant 50 in aq1

[23] Replace all the values below 60 of Temp with global constant 60 in aq2

[24] Replace the month numbers in the column Month in aq by name of the month. (Ex: Replace 5 with May). (Hint: use gsub() function. aq$Month <- gsub(5,"May",aq$Month))

[25] Create a new logical attribute Solar.Danger in aq by filling it's value with TRUE if the value in the attribute Solar.R is greater than 100, other with FALSE

[26] Discretize the values in Temp of aq to “Low”, “Medium” and “High”
[27] What does cut() function do?
[28] Create a numeric vector brks containing values 0, 50, 100, 200, 250, 300 and 350. Divide the range of Solar.Rinto intervals and recode the values in Solar.R according to which interval they fall using the vector brks.
aq$Solar.R=cut(aq$Solar.R,breaks=brks,include.lowest=TRUE)

```
# KHAN MOHD OWAIS RAZA
# 20BCD7138

# 1. Print the dataset airquality
print(airquality)

# 2. Print the structure of the dataset airquality
str(airquality)

# 3. Print the summary of all the variables of the dataset
airquality
summary(airquality)

# 4. How many variables (columns) are in the dataset airquality?
num_variables <- ncol(airquality)
print(num_variables)

# 5. How many observations (rows) are in the dataset airquality?
num_observations <- nrow(airquality)
print(num_observations)

# 6. The summary() function displays various descriptive
statistics for each variable, such as minimum, 1st quartile,
median, mean, 3rd quartile, maximum, and the number of missing
values.

# 7. Quartiles divide a dataset into four equal parts. They are
calculated as the values that divide the data into quarters. The
```

```
first quartile (Q1) represents the 25th percentile, and the third
quartile (Q3) represents the 75th percentile.
# We can find quartiles using the quantile() function in R.

# 8. Find the 1st and 3rd quartiles
q1 <- quantile(airquality$Ozone, 0.25, na.rm = TRUE)
q3 <- quantile(airquality$Ozone, 0.75, na.rm = TRUE)
print(q1)
print(q3)

# 9. Copy the dataset airquality to aq
aq <- airquality

# 10. Print the dataset aq
print(aq)

# 11. Print the structure of the dataset aq
str(aq)

# 12. Print the summary of all the variables of the dataset aq
summary(aq)

# 13. Print top 6 observations
print(head(aq, 6))

# 14. Print last 6 observations
print(tail(aq, 6))

# 15. Replace NA values in the "Ozone" attribute in aq with zero
aq$Ozone[is.na(aq$Ozone)] <- 0

# 16. Print the summary of all the variables of the dataset aq
summary(aq)

# 17. Replace NA values in the "Ozone" attribute in aq with the
mean of the remaining values
mean_ozone <- mean(aq$Ozone, na.rm = TRUE)
```

```
aq$Ozone[is.na(aq$Ozone)] <- mean_ozone

# 18. Print the summary of the dataset aq
summary(aq)

# 19. Copy the dataset airquality to aq1. Replace NA values in
the "Ozone" attribute in aq1 with the median of the remaining
values.
aq1 <- airquality
median_ozone <- median(aq1$Ozone, na.rm = TRUE)
aq1$Ozone[is.na(aq1$Ozone)] <- median_ozone
# Print the summary of the dataset aq1
summary(aq1)

# 20. Copy the dataset airquality to aq2. Replace NA values in
the "Ozone" attribute in aq2 with the mode of the remaining
values.
aq2 <- airquality
mode_ozone <-
as.numeric(names(table(aq2$Ozone)))[which.max(table(aq2$Ozone))]
aq2$Ozone[is.na(aq2$Ozone)] <- mode_ozone
# Print the summary of the dataset aq2
summary(aq2)

# 21. Repeat the above operations for the "Solar.R" attribute.

# 22. Replace all the values of "Temp" with the global constant
50 in aq1
aq1$Temp <- 50

# 23. Replace all values below 60 of "Temp" with the global
constant 60 in aq2
aq2$Temp[aq2$Temp < 60] <- 60

# 24. Replace the month numbers in the "Month" column in aq by
the name of the month using gsub() function.
aq$Month <- gsub("5", "May", aq$Month)
```

```
# 25. Create a new logical attribute "Solar.Danger" in aq by
filling it with TRUE if the value in the "Solar.R" attribute is
greater than 100, otherwise FALSE.
aq$Solar.Danger <- aq$Solar.R > 100

# 26. Discretize the values in "Temp" of aq into "Low", "Medium",
and "High".
aq$Temp <- cut(aq$Temp, breaks = c(-Inf, 60, 80, Inf), labels =
c("Low", "Medium", "High"))

# 27. The cut() function in R is used to divide a continuous
variable into intervals or groups (discretization).

# 28. Create a numeric vector brks containing values 0, 50, 100,
# 200, 250, 300, and 350. Divide the range of "Solar.R" into
# intervals and recode the values in "Solar.R" according to which
# interval they fall using the vector brks.
brks <- c(0, 50, 100, 200, 250, 300, 350)
aq$Solar.R <- cut(aq$Solar.R, breaks = brks, include.lowest =
TRUE)
```

```
> # KHAN MOHD OWAIS RAZA
> # 20BCD7138
>
> # 1. Print the dataset airquality
> print(airquality)
    Ozone Solar.R Wind Temp Month Day
1      41     190  7.4   67     5   1
2      36     118  8.0   72     5   2
3      12     149 12.6   74     5   3
4      18     313 11.5   62     5   4
5      NA      NA 14.3   56     5   5
6      28      NA 14.9   66     5   6
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
10     NA     194  8.6   69     5  10
11      7      NA  6.9   74     5  11
12     16     256  9.7   69     5  12
13     11     290  9.2   66     5  13
14     14     274 10.9   68     5  14
15     18      65 13.2   58     5  15
16     14     334 11.5   64     5  16
17     34     307 12.0   66     5  17
18      6      78 18.4   57     5  18
```

19	30	322	11.5	68	5	19
20	11	44	9.7	62	5	20
21	1	8	9.7	59	5	21
22	11	320	16.6	73	5	22
23	4	25	9.7	61	5	23
24	32	92	12.0	61	5	24
25	NA	66	16.6	57	5	25
26	NA	266	14.9	58	5	26
27	NA	NA	8.0	57	5	27
28	23	13	12.0	67	5	28
29	45	252	14.9	81	5	29
30	115	223	5.7	79	5	30
31	37	279	7.4	76	5	31
32	NA	286	8.6	78	6	1
33	NA	287	9.7	74	6	2
34	NA	242	16.1	67	6	3
35	NA	186	9.2	84	6	4
36	NA	220	8.6	85	6	5
37	NA	264	14.3	79	6	6
38	29	127	9.7	82	6	7
39	NA	273	6.9	87	6	8
40	71	291	13.8	90	6	9
41	39	323	11.5	87	6	10
42	NA	259	10.9	93	6	11
43	NA	250	9.2	92	6	12
44	23	148	8.0	82	6	13
45	NA	332	13.8	80	6	14
46	NA	322	11.5	79	6	15
47	21	191	14.9	77	6	16
48	37	284	20.7	72	6	17
49	20	37	9.2	65	6	18
50	12	120	11.5	73	6	19
51	13	137	10.3	76	6	20
52	NA	150	6.3	77	6	21
53	NA	59	1.7	76	6	22
54	NA	91	4.6	76	6	23
55	NA	250	6.3	76	6	24
56	NA	135	8.0	75	6	25
57	NA	127	8.0	78	6	26
58	NA	47	10.3	73	6	27
59	NA	98	11.5	80	6	28
60	NA	31	14.9	77	6	29
61	NA	138	8.0	83	6	30
62	135	269	4.1	84	7	1
63	49	248	9.2	85	7	2
64	32	236	9.2	81	7	3
65	NA	101	10.9	84	7	4
66	64	175	4.6	83	7	5
67	40	314	10.9	83	7	6
68	77	276	5.1	88	7	7
69	97	267	6.3	92	7	8
70	97	272	5.7	92	7	9
71	85	175	7.4	89	7	10
72	NA	139	8.6	82	7	11
73	10	264	14.3	73	7	12
74	27	175	14.9	81	7	13
75	NA	291	14.9	91	7	14
76	7	48	14.3	80	7	15
77	48	260	6.9	81	7	16
78	35	274	10.3	82	7	17

79	61	285	6.3	84	7	18
80	79	187	5.1	87	7	19
81	63	220	11.5	85	7	20
82	16	7	6.9	74	7	21
83	NA	258	9.7	81	7	22
84	NA	295	11.5	82	7	23
85	80	294	8.6	86	7	24
86	108	223	8.0	85	7	25
87	20	81	8.6	82	7	26
88	52	82	12.0	86	7	27
89	82	213	7.4	88	7	28
90	50	275	7.4	86	7	29
91	64	253	7.4	83	7	30
92	59	254	9.2	81	7	31
93	39	83	6.9	81	8	1
94	9	24	13.8	81	8	2
95	16	77	7.4	82	8	3
96	78	NA	6.9	86	8	4
97	35	NA	7.4	85	8	5
98	66	NA	4.6	87	8	6
99	122	255	4.0	89	8	7
100	89	229	10.3	90	8	8
101	110	207	8.0	90	8	9
102	NA	222	8.6	92	8	10
103	NA	137	11.5	86	8	11
104	44	192	11.5	86	8	12
105	28	273	11.5	82	8	13
106	65	157	9.7	80	8	14
107	NA	64	11.5	79	8	15
108	22	71	10.3	77	8	16
109	59	51	6.3	79	8	17
110	23	115	7.4	76	8	18
111	31	244	10.9	78	8	19
112	44	190	10.3	78	8	20
113	21	259	15.5	77	8	21
114	9	36	14.3	72	8	22
115	NA	255	12.6	75	8	23
116	45	212	9.7	79	8	24
117	168	238	3.4	81	8	25
118	73	215	8.0	86	8	26
119	NA	153	5.7	88	8	27
120	76	203	9.7	97	8	28
121	118	225	2.3	94	8	29
122	84	237	6.3	96	8	30
123	85	188	6.3	94	8	31
124	96	167	6.9	91	9	1
125	78	197	5.1	92	9	2
126	73	183	2.8	93	9	3
127	91	189	4.6	93	9	4
128	47	95	7.4	87	9	5
129	32	92	15.5	84	9	6
130	20	252	10.9	80	9	7
131	23	220	10.3	78	9	8
132	21	230	10.9	75	9	9
133	24	259	9.7	73	9	10
134	44	236	14.9	81	9	11
135	21	259	15.5	76	9	12
136	28	238	6.3	77	9	13
137	9	24	10.9	71	9	14
138	13	112	11.5	71	9	15

```
139    46     237  6.9   78     9  16
140    18     224 13.8   67     9  17
141    13      27 10.3   76     9  18
142    24     238 10.3   68     9  19
143    16     201  8.0   82     9  20
144    13     238 12.6   64     9  21
145    23      14  9.2   71     9  22
146    36     139 10.3   81     9  23
147     7      49 10.3   69     9  24
148    14      20 16.6   63     9  25
149    30     193  6.9   70     9  26
150    NA     145 13.2   77     9  27
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
>
> # 2. Print the structure of the dataset airquality
> str(airquality)
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
>
> # 3. Print the summary of all the variables of the dataset airquality
> summary(airquality)
     Ozone           Solar.R           Wind             Temp           Month            Day
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
 NA's   :37       NA's   :7
>
> # 4. How many variables (columns) are in the dataset airquality?
> num_variables <- ncol(airquality)
> print(num_variables)
[1] 6
>
> # 5. How many observations (rows) are in the dataset airquality?
> num_observations <- nrow(airquality)
> print(num_observations)
[1] 153
>
> # 6. The summary() function displays various descriptive statistics for each variable, such as minimum,
1st quartile, median, mean, 3rd quartile, maximum, and the number of missing values.
>
> # 7. Quartiles divide a dataset into four equal parts. They are calculated as the values that divide the
data into quarters. The first quartile (Q1) represents the 25th percentile, and the third quartile (Q3)
represents the 75th percentile.
> # We can find quartiles using the quantile() function in R.
>
> # 8. Find the 1st and 3rd quartiles
> q1 <- quantile(airquality$Ozone, 0.25, na.rm = TRUE)
> q3 <- quantile(airquality$Ozone, 0.75, na.rm = TRUE)
> print(q1)
25%
```

```
 18
> print(q3)
  75%
63.25
>
> # 9. Copy the dataset airquality to aq
> aq <- airquality
>
> # 10. Print the dataset aq
> print(aq)
    Ozone Solar.R Wind Temp Month Day
1      41     190  7.4   67     5   1
2      36     118  8.0   72     5   2
3      12     149 12.6   74     5   3
4      18     313 11.5   62     5   4
5      NA      NA 14.3   56     5   5
6      28      NA 14.9   66     5   6
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
10     NA     194  8.6   69     5  10
11      7      NA  6.9   74     5  11
12     16     256  9.7   69     5  12
13     11     290  9.2   66     5  13
14     14     274 10.9   68     5  14
15     18      65 13.2   58     5  15
16     14     334 11.5   64     5  16
17     34     307 12.0   66     5  17
18      6      78 18.4   57     5  18
19     30     322 11.5   68     5  19
20     11      44  9.7   62     5  20
21      1       8  9.7   59     5  21
22     11     320 16.6   73     5  22
23      4      25  9.7   61     5  23
24     32      92 12.0   61     5  24
25     NA      66 16.6   57     5  25
26     NA     266 14.9   58     5  26
27     NA      NA  8.0   57     5  27
28     23      13 12.0   67     5  28
29     45     252 14.9   81     5  29
30    115     223  5.7   79     5  30
31     37     279  7.4   76     5  31
32     NA     286  8.6   78     6   1
33     NA     287  9.7   74     6   2
34     NA     242 16.1   67     6   3
35     NA     186  9.2   84     6   4
36     NA     220  8.6   85     6   5
37     NA     264 14.3   79     6   6
38     29     127  9.7   82     6   7
39     NA     273  6.9   87     6   8
40     71     291 13.8   90     6   9
41     39     323 11.5   87     6  10
42     NA     259 10.9   93     6  11
43     NA     250  9.2   92     6  12
44     23     148  8.0   82     6  13
45     NA     332 13.8   80     6  14
46     NA     322 11.5   79     6  15
47     21     191 14.9   77     6  16
48     37     284 20.7   72     6  17
49     20      37  9.2   65     6  18
```

50	12	120	11.5	73	6	19
51	13	137	10.3	76	6	20
52	NA	150	6.3	77	6	21
53	NA	59	1.7	76	6	22
54	NA	91	4.6	76	6	23
55	NA	250	6.3	76	6	24
56	NA	135	8.0	75	6	25
57	NA	127	8.0	78	6	26
58	NA	47	10.3	73	6	27
59	NA	98	11.5	80	6	28
60	NA	31	14.9	77	6	29
61	NA	138	8.0	83	6	30
62	135	269	4.1	84	7	1
63	49	248	9.2	85	7	2
64	32	236	9.2	81	7	3
65	NA	101	10.9	84	7	4
66	64	175	4.6	83	7	5
67	40	314	10.9	83	7	6
68	77	276	5.1	88	7	7
69	97	267	6.3	92	7	8
70	97	272	5.7	92	7	9
71	85	175	7.4	89	7	10
72	NA	139	8.6	82	7	11
73	10	264	14.3	73	7	12
74	27	175	14.9	81	7	13
75	NA	291	14.9	91	7	14
76	7	48	14.3	80	7	15
77	48	260	6.9	81	7	16
78	35	274	10.3	82	7	17
79	61	285	6.3	84	7	18
80	79	187	5.1	87	7	19
81	63	220	11.5	85	7	20
82	16	7	6.9	74	7	21
83	NA	258	9.7	81	7	22
84	NA	295	11.5	82	7	23
85	80	294	8.6	86	7	24
86	108	223	8.0	85	7	25
87	20	81	8.6	82	7	26
88	52	82	12.0	86	7	27
89	82	213	7.4	88	7	28
90	50	275	7.4	86	7	29
91	64	253	7.4	83	7	30
92	59	254	9.2	81	7	31
93	39	83	6.9	81	8	1
94	9	24	13.8	81	8	2
95	16	77	7.4	82	8	3
96	78	NA	6.9	86	8	4
97	35	NA	7.4	85	8	5
98	66	NA	4.6	87	8	6
99	122	255	4.0	89	8	7
100	89	229	10.3	90	8	8
101	110	207	8.0	90	8	9
102	NA	222	8.6	92	8	10
103	NA	137	11.5	86	8	11
104	44	192	11.5	86	8	12
105	28	273	11.5	82	8	13
106	65	157	9.7	80	8	14
107	NA	64	11.5	79	8	15
108	22	71	10.3	77	8	16
109	59	51	6.3	79	8	17

```
110    23     115  7.4   76     8  18
111    31     244 10.9   78     8  19
112    44     190 10.3   78     8  20
113    21     259 15.5   77     8  21
114     9      36 14.3   72     8  22
115    NA     255 12.6   75     8  23
116    45     212  9.7   79     8  24
117   168     238  3.4   81     8  25
118    73     215  8.0   86     8  26
119    NA     153  5.7   88     8  27
120    76     203  9.7   97     8  28
121   118     225  2.3   94     8  29
122    84     237  6.3   96     8  30
123    85     188  6.3   94     8  31
124    96     167  6.9   91     9   1
125    78     197  5.1   92     9   2
126    73     183  2.8   93     9   3
127    91     189  4.6   93     9   4
128    47      95  7.4   87     9   5
129    32      92 15.5   84     9   6
130    20     252 10.9   80     9   7
131    23     220 10.3   78     9   8
132    21     230 10.9   75     9   9
133    24     259  9.7   73     9  10
134    44     236 14.9   81     9  11
135    21     259 15.5   76     9  12
136    28     238  6.3   77     9  13
137     9      24 10.9   71     9  14
138    13     112 11.5   71     9  15
139    46     237  6.9   78     9  16
140    18     224 13.8   67     9  17
141    13      27 10.3   76     9  18
142    24     238 10.3   68     9  19
143    16     201  8.0   82     9  20
144    13     238 12.6   64     9  21
145    23      14  9.2   71     9  22
146    36     139 10.3   81     9  23
147     7      49 10.3   69     9  24
148    14      20 16.6   63     9  25
149    30     193  6.9   70     9  26
150    NA     145 13.2   77     9  27
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
>
> # 11. Print the structure of the dataset aq
> str(aq)
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
>
> # 12. Print the summary of all the variables of the dataset aq
> summary(aq)
     Ozone           Solar.R           Wind             Temp           Month            Day
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
```

```
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
 NA's   :37       NA's   :7
>
> # 13. Print top 6 observations
> print(head(aq, 6))
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
>
> # 14. Print last 6 observations
> print(tail(aq, 6))
    Ozone Solar.R Wind Temp Month Day
148    14      20 16.6   63     9  25
149    30     193  6.9   70     9  26
150    NA     145 13.2   77     9  27
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
>
> # 15. Replace NA values in the "Ozone" attribute in aq with zero
> aq$Ozone[is.na(aq$Ozone)] <- 0
>
> # 16. Print the summary of all the variables of the dataset aq
> summary(aq)
     Ozone           Solar.R           Wind             Temp           Month            Day
 Min.   :  0.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.:  4.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 21.00   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 31.94   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 46.00   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
                  NA's   :7
>
> # 17. Replace NA values in the "Ozone" attribute in aq with the mean of the remaining values
> mean_ozone <- mean(aq$Ozone, na.rm = TRUE)
> aq$Ozone[is.na(aq$Ozone)] <- mean_ozone
>
> # 18. Print the summary of the dataset aq
> summary(aq)
     Ozone           Solar.R           Wind             Temp           Month            Day
 Min.   :  0.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.:  4.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 21.00   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 31.94   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 46.00   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
                  NA's   :7
>
> # 19. Copy the dataset airquality to aq1. Replace NA values in the "Ozone" attribute in aq1 with the
median of the remaining values.
> aq1 <- airquality
> median_ozone <- median(aq1$Ozone, na.rm = TRUE)
> aq1$Ozone[is.na(aq1$Ozone)] <- median_ozone
```

```
> # Print the summary of the dataset aq1
> summary(aq1)
     Ozone           Solar.R           Wind             Temp           Month            Day
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.: 21.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 39.56   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 46.00   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
                  NA's   :7
>
> # 20. Copy the dataset airquality to aq2. Replace NA values in the "Ozone" attribute in aq2 with the
mode of the remaining values.
> aq2 <- airquality
> mode_ozone <- as.numeric(names(table(aq2$Ozone)))[which.max(table(aq2$Ozone))]
> aq2$Ozone[is.na(aq2$Ozone)] <- mode_ozone
> # Print the summary of the dataset aq2
> summary(aq2)
     Ozone           Solar.R           Wind             Temp           Month            Day
 Min.   :  1.0   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000   Min.   : 1.0
 1st Qu.: 21.0   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000   1st Qu.: 8.0
 Median : 23.0   Median :205.0   Median : 9.700   Median :79.00   Median :7.000   Median :16.0
 Mean   : 37.5   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993   Mean   :15.8
 3rd Qu.: 46.0   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000   3rd Qu.:23.0
 Max.   :168.0   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000   Max.   :31.0
                 NA's   :7
>
> # 21. Repeat the above operations for the "Solar.R" attribute.
> # Replace NA values in the "Solar.R" attribute in aq1 with zero
> aq1$Solar.R[is.na(aq1$Solar.R)] <- 0
> aq1$Solar.R
  [1] 190 118 149 313   0   0 299  99  19 194   0 256 290 274  65 334 307  78 322  44   8 320  25  92  66
266   0  13 252 223 279 286 287 242 186 220 264
 [38] 127 273 291 323 259 250 148 332 322 191 284  37 120 137 150  59  91 250 135 127  47  98  31 138 269
248 236 101 175 314 276 267 272 175 139 264 175
 [75] 291  48 260 274 285 187 220   7 258 295 294 223  81  82 213 275 253 254  83  24  77   0   0   0 255
229 207 222 137 192 273 157  64  71  51 115 244
[112] 190 259  36 255 212 238 215 153 203 225 237 188 167 197 183 189  95  92 252 220 230 259 236 259 238
24 112 237 224  27 238 201 238  14 139  49  20
[149] 193 145 191 131 223
> aq1$Solar.R[is.na(aq1$Solar.R)]
numeric(0)
> # Replace NA values in the "Solar.R" attribute in aq2 with the mean of the remaining values
> mean_solar <- mean(aq2$Solar.R, na.rm = TRUE)
> aq2$Solar.R[is.na(aq2$Solar.R)] <- mean_solar
> aq2$Solar.R
  [1] 190.0000 118.0000 149.0000 313.0000 185.9315 185.9315 299.0000  99.0000  19.0000 194.0000 185.9315
256.0000 290.0000 274.0000  65.0000 334.0000
 [17] 307.0000  78.0000 322.0000  44.0000   8.0000 320.0000  25.0000  92.0000  66.0000 266.0000 185.9315
13.0000 252.0000 223.0000 279.0000 286.0000
 [33] 287.0000 242.0000 186.0000 220.0000 264.0000 127.0000 273.0000 291.0000 323.0000 259.0000 250.0000
148.0000 332.0000 322.0000 191.0000 284.0000
 [49]  37.0000 120.0000 137.0000 150.0000  59.0000  91.0000 250.0000 135.0000 127.0000  47.0000  98.0000
31.0000 138.0000 269.0000 248.0000 236.0000
 [65] 101.0000 175.0000 314.0000 276.0000 267.0000 272.0000 175.0000 139.0000 264.0000 175.0000 291.0000
48.0000 260.0000 274.0000 285.0000 187.0000
 [81] 220.0000   7.0000 258.0000 295.0000 294.0000 223.0000  81.0000  82.0000 213.0000 275.0000 253.0000
254.0000  83.0000  24.0000  77.0000 185.9315
 [97] 185.9315 185.9315 255.0000 229.0000 207.0000 222.0000 137.0000 192.0000 273.0000 157.0000  64.0000
71.0000  51.0000 115.0000 244.0000 190.0000
```

```
[113] 259.0000  36.0000 255.0000 212.0000 238.0000 215.0000 153.0000 203.0000 225.0000 237.0000 188.0000
167.0000 197.0000 183.0000 189.0000  95.0000
[129]  92.0000 252.0000 220.0000 230.0000 259.0000 236.0000 259.0000 238.0000  24.0000 112.0000 237.0000
224.0000  27.0000 238.0000 201.0000 238.0000
[145]  14.0000 139.0000  49.0000  20.0000 193.0000 145.0000 191.0000 131.0000 223.0000
> aq2$Solar.R[is.na(aq2$Solar.R)]
numeric(0)
>
> # 22. Replace all the values of "Temp" with the global constant 50 in aq1
> aq1$Temp <- 50
> aq1$Temp
  [1] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
 [50] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
 [99] 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
50 50 50 50 50 50 50 50 50 50 50 50 50 50 50
[148] 50 50 50 50 50 50
>
> # 23. Replace all values below 60 of "Temp" with the global constant 60 in aq2
> aq2$Temp[aq2$Temp < 60] <- 60
> aq2$Temp
  [1] 67 72 74 62 60 66 65 60 61 69 74 69 66 68 60 64 66 60 68 62 60 73 61 61 60 60 60 67 81 79 76 78 74
67 84 85 79 82 87 90 87 93 92 82 80 79 77 72 65
 [50] 73 76 77 76 76 76 75 78 73 80 77 83 84 85 81 84 83 83 88 92 92 89 82 73 81 91 80 81 82 84 87 85 74
81 82 86 85 82 86 88 86 83 81 81 81 82 86 85 87
 [99] 89 90 90 92 86 86 82 80 79 77 79 76 78 78 77 72 75 79 81 86 88 97 94 96 94 91 92 93 93 87 84 80 78
75 73 81 76 77 71 71 78 67 76 68 82 64 71 81 69
[148] 63 70 77 75 76 68
> aq2$Temp[aq2$Temp < 60]
numeric(0)
>
> # 24. Replace the month numbers in the "Month" column in aq by the name of the month using gsub()
function.
> aq$Month <- gsub("5", "May", aq$Month)
> aq$Month
  [1] "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May" "May"
"May" "May" "May" "May" "May" "May" "May" "May"
 [25] "May" "May" "May" "May" "May" "May" "May" "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"
"6"   "6"   "6"   "6"   "6"   "6"   "6"
 [49] "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "6"   "7"   "7"   "7"   "7"
"7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"
 [73] "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"   "7"
"7"   "7"   "7"   "8"   "8"   "8"   "8"
 [97] "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"   "8"
"8"   "8"   "8"   "8"   "8"   "8"   "8"
[121] "8"   "8"   "8"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"
"9"   "9"   "9"   "9"   "9"   "9"   "9"
[145] "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"   "9"
>
> # 25. Create a new logical attribute "Solar.Danger" in aq by filling it with TRUE if the value in the
"Solar.R" attribute is greater than 100, otherwise FALSE.
> aq$Solar.Danger <- aq$Solar.R > 100
Warning message:
In Ops.factor(aq$Solar.R, 100) : ‘>’ not meaningful for factors
> aq$Solar.Danger
  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
 [50] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
```

```
 [99] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
[148] NA NA NA NA NA NA
>
> # 26. Discretize the values in "Temp" of aq into "Low", "Medium", and "High".
> aq$Temp <- cut(aq$Temp, breaks = c(-Inf, 60, 80, Inf), labels = c("Low", "Medium", "High"))
Error in cut.default(aq$Temp, breaks = c(-Inf, 60, 80, Inf), labels = c("Low",  :
  'x' must be numeric
> aq$Temp
  [1] Medium Medium Medium Medium Low    Medium Medium Low    Medium Medium Medium Medium Medium Medium
Low    Medium Medium Low    Medium Medium Low
 [22] Medium Medium Medium Low    Low    Low    Medium High   Medium Medium Medium Medium Medium High
High   Medium High   High   High   High   High
 [43] High   High   Medium Medium Medium Medium Medium Medium Medium Medium Medium Medium Medium Medium
Medium Medium Medium Medium High   High   High
 [64] High   High   High   High   High   High   High   High   High   Medium High   High   Medium High
High   High   High   High   Medium High   High
 [85] High   High   High   High   High   High   High   High   High   High   High   High   High   High
High   High   High   High   High   High   High
[106] Medium Medium Medium Medium Medium Medium Medium Medium Medium Medium Medium High   High   High
High   High   High   High   High   High   High
[127] High   High   High   Medium Medium Medium Medium High   Medium Medium Medium Medium Medium Medium
Medium Medium Medium High   Medium Medium High   Medium
[148] Medium Medium Medium Medium Medium Medium
Levels: Low Medium High
>
> # 27. The cut() function in R is used to divide a continuous variable into intervals or groups
(discretization).
>
> # 28. Create a numeric vector brks containing values 0, 50, 100, 200, 250, 300, and 350. Divide the
range of "Solar.R" into intervals and recode the values in "Solar.R" according to which interval they fall
using the vector brks.
> brks <- c(0, 50, 100, 200, 250, 300, 350)
> aq$Solar.R <- cut(aq$Solar.R, breaks = brks, include.lowest = TRUE)
Error in cut.default(aq$Solar.R, breaks = brks, include.lowest = TRUE) :
  'x' must be numeric
> brks
[1]   0  50 100 200 250 300 350
> aq$Solar.R
  [1] (100,200] (100,200] (100,200] (300,350] <NA>      <NA>      (250,300] (50,100]  [0,50]    (100,200]
<NA>      (250,300] (250,300] (250,300]
 [15] (50,100]  (300,350] (300,350] (50,100]  (300,350] [0,50]    [0,50]    (300,350] [0,50]    (50,100]
(50,100]  (250,300] <NA>      [0,50]
 [29] (250,300] (200,250] (250,300] (250,300] (250,300] (200,250] (100,200] (200,250] (250,300] (100,200]
(250,300] (250,300] (300,350] (250,300]
 [43] (200,250] (100,200] (300,350] (300,350] (100,200] (250,300] [0,50]    (100,200] (100,200] (100,200]
(50,100]  (50,100]  (200,250] (100,200]
 [57] (100,200] [0,50]    (50,100]  [0,50]    (100,200] (250,300] (200,250] (200,250] (100,200] (100,200]
(300,350] (250,300] (250,300] (250,300]
 [71] (100,200] (100,200] (250,300] (100,200] (250,300] [0,50]    (250,300] (250,300] (250,300] (100,200]
(200,250] [0,50]    (250,300] (250,300]
 [85] (250,300] (200,250] (50,100]  (50,100]  (200,250] (250,300] (250,300] (250,300] (50,100]  [0,50]
(50,100]  <NA>      <NA>      <NA>
 [99] (250,300] (200,250] (200,250] (200,250] (100,200] (100,200] (250,300] (100,200] (50,100]  (50,100]
(50,100]  (100,200] (200,250] (100,200]
[113] (250,300] [0,50]    (250,300] (200,250] (200,250] (200,250] (100,200] (200,250] (200,250] (200,250]
(100,200] (100,200] (100,200] (100,200]
[127] (100,200] (50,100]  (50,100]  (250,300] (200,250] (200,250] (250,300] (200,250] (250,300] (200,250]
[0,50]    (100,200] (200,250] (200,250]
```

```
[141] [0,50]    (200,250] (200,250] (200,250] [0,50]    (100,200] [0,50]    [0,50]    (100,200] (100,200]
(100,200] (100,200] (200,250]
Levels: [0,50] (50,100] (100,200] (200,250] (250,300] (300,350]
```

The output was too large to be pasted so I pasted the output text.

Q.3 Create the dataframes to merge:
[1] buildings <- data.frame(location=c(1, 2, 3), name=c("building1", "building2", "building3"))
[2] data <- data.frame(survey=c(1,1,1,2,2,2), location=c(1,2,3,2,3,1), efficiency=c(51,64,70,71,80,58))
[3] The dataframes, buildings and data have a common key variable called, "location". Use the merge() function to merge the two dataframes by "location", into a new dataframe, "buildingStats".

```
# KHAN MOHD OWAIS RAZA
# 20BCD7138
# [1] Create the "buildings" dataframe
buildings <- data.frame(location = c(1, 2, 3), name =
c("building1", "building2", "building3"))
buildings

# [2] Create the "data" dataframe
data <- data.frame(survey = c(1, 1, 1, 2, 2, 2), location = c(1,
2, 3, 2, 3, 1), efficiency = c(51, 64, 70, 71, 80, 58))
data

# [3] Merge the dataframes by "location" into a new dataframe
"buildingStats"
buildingStats <- merge(buildings, data, by = "location")
buildingStats
```

```
> # KHAN MOHD OWAIS RAZA
> # 20BCD7138
> # [1] Create the "buildings" dataframe
> buildings <- data.frame(location = c(1, 2, 3), name = c("building1",
"building2", "building3"))
> buildings
  location      name
1        1 building1
2        2 building2
3        3 building3
>
> # [2] Create the "data" dataframe
> data <- data.frame(survey = c(1, 1, 1, 2, 2, 2), location = c(1, 2,
3, 2, 3, 1), efficiency = c(51, 64, 70, 71, 80, 58))
> data
  survey location efficiency
1      1        1         51
2      1        2         64
3      1        3         70
4      2        2         71
5      2        3         80
6      2        1         58
>
> # [3] Merge the dataframes by "location" into a new dataframe
"buildingStats"
> buildingStats <- merge(buildings, data, by = "location")
> buildingStats
  location      name survey efficiency
1        1 building1      1         51
2        1 building1      2         58
3        2 building2      1         64
4        2 building2      2         71
5        3 building3      1         70
6        3 building3      2         80
```

```
> # KHAN MOHD OWAIS RAZA
> # 20BCD7138
> # [1] Create the "buildings" dataframe
> buildings <- data.frame(location = c(1, 2, 3), name = c("building1", "building2", "building3"))
> buildings
  location      name
1        1 building1
2        2 building2
3        3 building3
>
> # [2] Create the "data" dataframe
> data <- data.frame(survey = c(1, 1, 1, 2, 2, 2), location = c(1, 2, 3, 2, 3, 1), efficiency = c(51, 64, 70, 71, 80, 58))
> data
  survey location efficiency
1      1        1         51
2      1        2         64
3      1        3         70
4      2        2         71
5      2        3         80
6      2        1         58
>
> # [3] Merge the dataframes by "location" into a new dataframe "buildingStats"
> buildingStats <- merge(buildings, data, by = "location")
> buildingStats
  location      name survey efficiency
1        1 building1      1         51
2        1 building1      2         58
3        2 building2      1         64
4        2 building2      2         71
5        3 building3      1         70
6        3 building3      2         80
```

Q.4] Give the dataframes different key variable names:

buildings <- data.frame(location=c(1, 2, 3), name=c("building1", "building2", "building3"))

data <- data.frame(survey=c(1,1,1,2,2,2), LocationID=c(1,2,3,2,3,1), efficiency=c(51,64,70,71,80,58))

The dataframes, buildings and data now have corresponding variables called, location, and LocationID. Use the merge() function to merge the columns of the two dataframes by the corresponding variables.

```
# KHAN MOHD OWAIS RAZA
# 20BCD7138
# Create the "buildings" dataframe
buildings <- data.frame(location = c(1, 2, 3), name =
c("building1", "building2", "building3"))
buildings

# Create the "data" dataframe
```

```
data <- data.frame(survey = c(1, 1, 1, 2, 2, 2), LocationID =
c(1, 2, 3, 2, 3, 1), efficiency = c(51, 64, 70, 71, 80, 58))
data

# Merge the columns of the dataframes by the corresponding
variables
buildingStats <- merge(buildings, data, by.x = "location", by.y =
"LocationID")
buildingStats
```

```
> # KHAN MOHD OWAIS RAZA
> # 20BCD7138
> # Create the "buildings" dataframe
> buildings <- data.frame(location = c(1, 2, 3), name = c("building1", "building2", "building3"))
> buildings
  location      name
1        1 building1
2        2 building2
3        3 building3
>
> # Create the "data" dataframe
> data <- data.frame(survey = c(1, 1, 1, 2, 2, 2), LocationID = c(1, 2, 3, 2, 3, 1), efficiency = c(51, 64, 70, 71, 80, 58))
> data
  survey LocationID efficiency
1      1          1         51
2      1          2         64
3      1          3         70
4      2          2         71
5      2          3         80
6      2          1         58
>
> # Merge the columns of the dataframes by the corresponding variables
> buildingStats <- merge(buildings, data, by.x = "location", by.y = "LocationID")
> buildingStats
  location      name survey efficiency
1        1 building1      1         51
2        1 building1      2         58
3        2 building2      1         64
4        2 building2      2         71
5        3 building3      1         70
6        3 building3      2         80
```

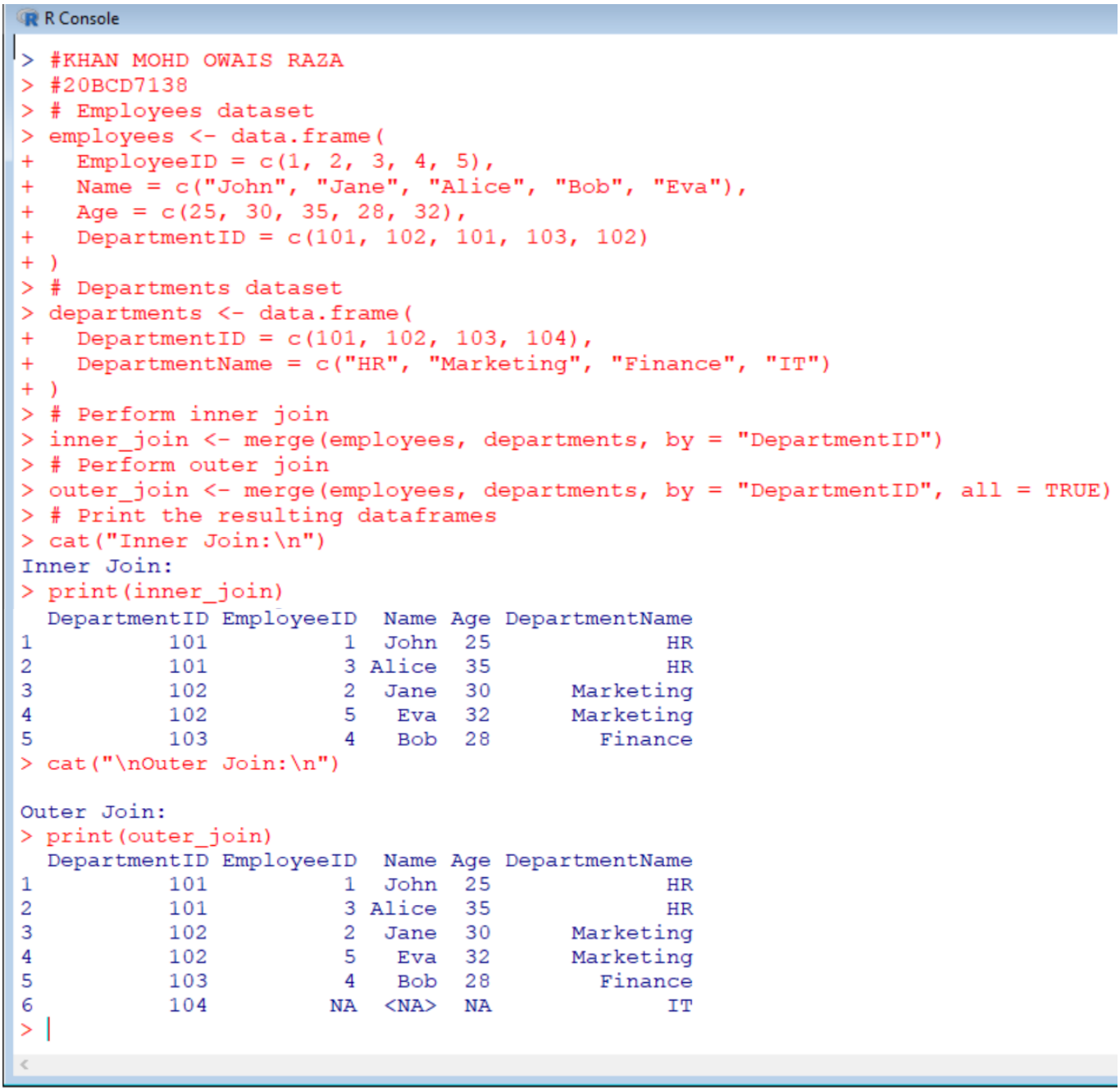
Q.5] Consider the following dataframes:-

```
# Employees dataset
employees <- data.frame(
  EmployeeID = c(1, 2, 3, 4, 5),
  Name = c("John", "Jane", "Alice", "Bob", "Eva"),
  Age = c(25, 30, 35, 28, 32),
  DepartmentID = c(101, 102, 101, 103, 102)
)
```

 # Departments dataset
departments <- data.frame(
DepartmentID = c(101, 102, 103, 104),
DepartmentName = c("HR", "Marketing", "Finance", "IT")
)

Perform innerjoin and outer join.

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Employees dataset
employees <- data.frame(
  EmployeeID = c(1, 2, 3, 4, 5),
  Name = c("John", "Jane", "Alice", "Bob", "Eva"),
  Age = c(25, 30, 35, 28, 32),
  DepartmentID = c(101, 102, 101, 103, 102)
)
# Departments dataset
departments <- data.frame(
  DepartmentID = c(101, 102, 103, 104),
  DepartmentName = c("HR", "Marketing", "Finance", "IT")
)
# Perform inner join
inner_join <- merge(employees, departments, by = "DepartmentID")
# Perform outer join
outer_join <- merge(employees, departments, by = "DepartmentID",
all = TRUE)
# Print the resulting dataframes
cat("Inner Join:\n")
print(inner_join)
cat("\nOuter Join:\n")
print(outer_join)
```



Q.6] Consider the following dataframes

 # Orders dataset
orders <- data.frame(
OrderID = c(1, 2, 3, 4, 5),
CustomerID = c(101, 102, 103, 101, 104),
Amount = c(100, 200, 150, 300, 250)
)
 # Customers dataset
customers <- data.frame(

CustomerID = c(101, 102, 103, 104, 105),
CustomerName = c("John", "Jane", "Alice", "Bob", "Eva")
)

Perform left join, right join and cross join

```
#KHAN MOHD OWAIS RAZA
#20BCD7138
# Orders dataset
orders <- data.frame(
  OrderID = c(1, 2, 3, 4, 5),
  CustomerID = c(101, 102, 103, 101, 104),
  Amount = c(100, 200, 150, 300, 250)
)
# Customers dataset
customers <- data.frame(
  CustomerID = c(101, 102, 103, 104, 105),
  CustomerName = c("John", "Jane", "Alice", "Bob", "Eva")
)
# Perform left join
left_join <- merge(orders, customers, by = "CustomerID", all.x =
TRUE)
# Perform right join
right_join <- merge(orders, customers, by = "CustomerID", all.y =
TRUE)
# Perform cross join
cross_join <- merge(orders, customers, by = NULL)
# Print the resulting dataframes
cat("Left Join:\n")
print(left_join)
cat("\nRight Join:\n")
print(right_join)
cat("\nCross Join:\n")
print(cross_join)
```

R Console

```
> #KHAN MOHD OWAIS RAZA
> #20BCD7138
> # Orders dataset
> orders <- data.frame(
+   OrderID = c(1, 2, 3, 4, 5),
+   CustomerID = c(101, 102, 103, 101, 104),
+   Amount = c(100, 200, 150, 300, 250)
+ )
> # Customers dataset
> customers <- data.frame(
+   CustomerID = c(101, 102, 103, 104, 105),
+   CustomerName = c("John", "Jane", "Alice", "Bob", "Eva")
+ )
> # Perform left join
> left_join <- merge(orders, customers, by = "CustomerID", all.x = TRUE)
> # Perform right join
> right_join <- merge(orders, customers, by = "CustomerID", all.y = TRUE)
> # Perform cross join
> cross_join <- merge(orders, customers, by = NULL)
> # Print the resulting dataframes
> cat("Left Join:\n")
Left Join:
> print(left_join)
  CustomerID OrderID Amount CustomerName
1        101       1    100         John
2        101       4    300         John
3        102       2    200         Jane
4        103       3    150        Alice
5        104       5    250          Bob
> cat("\nRight Join:\n")

Right Join:
> print(right_join)
  CustomerID OrderID Amount CustomerName
1        101       1    100         John
2        101       4    300         John
3        102       2    200         Jane
4        103       3    150        Alice
5        104       5    250          Bob
6        105      NA     NA          Eva
> cat("\nCross Join:\n")

Cross Join:
> print(cross_join)
   OrderID CustomerID.x Amount CustomerID.y CustomerName
1        1          101    100          101         John
2        2          102    200          101         John
3        3          103    150          101         John
4        4          101    300          101         John
5        5          104    250          101         John
6        1          101    100          102         Jane
7        2          102    200          102         Jane
```

8	3	103	150	102	Jane
9	4	101	300	102	Jane
10	5	104	250	102	Jane
11	1	101	100	103	Alice
12	2	102	200	103	Alice
13	3	103	150	103	Alice
14	4	101	300	103	Alice
15	5	104	250	103	Alice
16	1	101	100	104	Bob
17	2	102	200	104	Bob
18	3	103	150	104	Bob
19	4	101	300	104	Bob
20	5	104	250	104	Bob
21	1	101	100	105	Eva
22	2	102	200	105	Eva
23	3	103	150	105	Eva
24	4	101	300	105	Eva
25	5	104	250	105	Eva

>