FDA Lab-6

KHAN MOHD OWAIS RAZA
20BCD7138

Q1. Create a vector as x <- c(9:20, 1:5, 3:7, 0:8)
1] Use duplicated() function to print the logical vector indicating the duplicate values present in x
2] Observe the output of duplicated(x, fromLast = TRUE)
3] What is the difference between duplicated(x) and duplicated(x,fromLast=TRUE)?
4] Extract duplicate elements from x
5] Extract unique elements from x
6] Print duplicate elements from x in different order (Hint: Use duplicated(x, fromLast = TRUE))
7] Extract unique elements from x in different order (Hint: Use duplicated(x, fromLast = TRUE))
8] Print the indices of duplicate elements
9] Print the indices of unique elements
10] How many unique elements are in x?
11] How many duplicate elements are in x?

```
# Create the vector x
x <- c(9:20, 1:5, 3:7, 0:8)
x
# 1] Use duplicated() function to
# print logical vector indicating duplicate values
dup_vector <- duplicated(x)
print(dup_vector)

# 2] Use duplicated(x, fromLast = TRUE) to observe the output
dup_vector_from_last <- duplicated(x, fromLast = TRUE)
print(dup_vector_from_last)

# 3] The difference between duplicated(x)
# and duplicated(x, fromLast = TRUE)
# is the direction of checking for duplicates.
# duplicated(x) checks for duplicates from the first occurrence,
```

```
# while duplicated(x, fromLast = TRUE) checks from the last
occurrence.

# 4] Extract duplicate elements from x
duplicate_elements <- x[dup_vector]
print(duplicate_elements)

# 5] Extract unique elements from x
unique_elements <- x[!dup_vector]
print(unique_elements)

# 6] Print duplicate elements from x in a different order
# duplicate_elements_reverse <- x[dup_vector_from_last]
print(duplicate_elements_reverse)

# 8] Extract unique elements from x in a different order
unique_elements_reverse <- x[!dup_vector_from_last]
print(unique_elements_reverse)

# 9] Print the indices of duplicate elements
duplicate_indices <- which(dup_vector)
print(duplicate_indices)

# 10] Print the indices of unique elements
unique_indices <- which(!dup_vector)
print(unique_indices)

# Count the number of unique elements in x
num_unique_elements <- length(unique_elements)
print(num_unique_elements)

# 11] Count the number of duplicate elements in x
num_duplicate_elements <- length(duplicate_elements)
print(num_duplicate_elements)
```

R Console

```
> # Create the vector x
> x <- c(9:20, 1:5, 3:7, 0:8)
> x
 [1]  9 10 11 12 13 14 15 16 17 18 19 20  1  2  3  4  5  3  4  5  6  7  0  1  2  3  4  5  6
[30]  7  8
> # 1] Use duplicated() function to
> # print logical vector indicating duplicate values
> dup_vector <- duplicated(x)
> print(dup_vector)
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[15] FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
[29]  TRUE  TRUE FALSE
>
> # 2] Use duplicated(x, fromLast = TRUE) to observe the output
> dup_vector_from_last <- duplicated(x, fromLast = TRUE)
> print(dup_vector_from_last)
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
[15]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
[29] FALSE FALSE FALSE
>
> # 3] The difference between duplicated(x)
> # and duplicated(x, fromLast = TRUE)
> # is the direction of checking for duplicates.
> # duplicated(x) checks for duplicates from the first occurrence,
> # while duplicated(x, fromLast = TRUE) checks from the last occurrence.
>
> # 4] Extract duplicate elements from x
> duplicate_elements <- x[dup_vector]
> print(duplicate_elements)
 [1] 3 4 5 1 2 3 4 5 6 7
>
> # 5] Extract unique elements from x
> unique_elements <- x[!dup_vector]
> print(unique_elements)
 [1]  9 10 11 12 13 14 15 16 17 18 19 20  1  2  3  4  5  6  7  0  8
>
> # 6] Print duplicate elements from x in a different order
> # duplicate_elements_reverse <- x[dup_vector_from_last]
> print(duplicate_elements_reverse)
 [1] 1 2 3 4 5 3 4 5 6 7
>
> # 8] Extract unique elements from x in a different order
> unique_elements_reverse <- x[!dup_vector_from_last]
> print(unique_elements_reverse)
 [1]  9 10 11 12 13 14 15 16 17 18 19 20  0  1  2  3  4  5  6  7  8
>
> # 9] Print the indices of duplicate elements
> duplicate_indices <- which(dup_vector)
> print(duplicate_indices)
 [1] 18 19 20 24 25 26 27 28 29 30
>
> # 10] Print the indices of unique elements
> unique_indices <- which(!dup_vector)
> print(unique_indices)
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 21 22 23 31
>
> # Count the number of unique elements in x
> num_unique_elements <- length(unique_elements)
> print(num_unique_elements)
[1] 21
>
> # 11] Count the number of duplicate elements in x
> num_duplicate_elements <- length(duplicate_elements)
> print(num_duplicate_elements)
[1] 10
>
```

Q.2 Create a dataframe df :
a <- c(rep("A", 3), rep("B", 3), rep("C",2)) b <- c(1,1,2,4,1,1,2,2)
df <-data.frame(a,b)

1] Use duplicated() function to print the logical
vector indicating the duplicate values present in dataframe "df"
2] Extract duplicate elements from dataframe "df"
3] Extract unique elements from dataframe "df"
4] Print the indices of duplicate elements
5] Print the indices of unique elements
6] How many unique elements are in dataframe "df"
7] How many duplicate elements are in dataframe "df"

```
# Create the dataframe df
a <- c(rep("A",3), rep("B",3), rep("C",2))
b <- c(1,1,2,4,1,1,2,2)
df <- data.frame(a,b)
df
# 1] Use duplicated() function to print
# logical vector indicating duplicate values
dup_vector <- duplicated(df)
print(dup_vector)

# 2] Extract duplicate elements from dataframe df
duplicate_rows <- df[dup_vector, ]
print(duplicate_rows)

# 3] Extract unique elements from dataframe df
unique_rows <- df[!dup_vector, ]
print(unique_rows)

# 4] Print the indices of duplicate elements
duplicate_indices <- which(dup_vector)
print(duplicate_indices)

# 5] Print the indices of unique elements
unique_indices <- which(!dup_vector)
```

```
print(unique_indices)

# 6] Count the number of unique elements in dataframe df
num_unique_elements <- length(unique_rows)
print(num_unique_elements)

# 7] Count the number of duplicate elements in dataframe df
num_duplicate_elements <- length(duplicate_rows)
print(num_duplicate_elements)
```

R Console

```
> # Create the dataframe df
> a <- c(rep("A",3), rep("B",3), rep("C",2))
> b <- c(1,1,2,4,1,1,2,2)
> df <- data.frame(a,b)
> df
  a b
1 A 1
2 A 1
3 A 2
4 B 4
5 B 1
6 B 1
7 C 2
8 C 2
> # 1] Use duplicated() function to print
> # logical vector indicating duplicate values
> dup_vector <- duplicated(df)
> print(dup_vector)
[1] FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE
>
> # 2] Extract duplicate elements from dataframe df
> duplicate_rows <- df[dup_vector, ]
> print(duplicate_rows)
  a b
2 A 1
6 B 1
8 C 2
>
> # 3] Extract unique elements from dataframe df
> unique_rows <- df[!dup_vector, ]
> print(unique_rows)
  a b
1 A 1
3 A 2
4 B 4
5 B 1
7 C 2
```

```
> # 4] Print the indices of duplicate elements
> duplicate_indices <- which(dup_vector)
> print(duplicate_indices)
[1] 2 6 8
>
> # 5] Print the indices of unique elements
> unique_indices <- which(!dup_vector)
> print(unique_indices)
[1] 1 3 4 5 7
>
> # 6] Count the number of unique elements in dataframe df
> num_unique_elements <- length(unique_rows)
> print(num_unique_elements)
[1] 2
>
> # 7] Count the number of duplicate elements in dataframe df
> num_duplicate_elements <- length(duplicate_rows)
> print(num_duplicate_elements)
[1] 2
> |
```

Q.3 Consider a dataset Fisher's Iris Dataset
1] Print the dataset iris
2] Print the structure of the dataset iris
3] Print the summary of all the variables of
the dataset iris (Hint: Use function summary())
4] How many of the variables (columns) are in the dataset iris?
5] How many observations (rows) are in the dataset iris?
6] Use duplicated() function to print the logical
vector indicating the duplicate values present in the dataset iris
7] Extract duplicate elements from the dataset iris
8] Extract unique elements from the dataset iris
9] Print the indices of duplicate elements in the dataset iris
10] Print the indices of unique elements in the dataset iris
11] How many unique elements are in the dataset iris?
12] How many duplicate elements are in the dataset iris?

```
# 1. Print the dataset iris
print(iris)
# 2. Print the structure of the dataset iris
str(iris)
# 3. Print the summary of all the variables of the dataset iris
summary(iris)
```

```
# 4. How many variables (columns) are in the dataset iris?
num_variables <- ncol(iris)
print(num_variables)
# 5. How many observations (rows) are in the dataset iris?
num_observations <- nrow(iris)
print(num_observations)
# 6. Use duplicated() function to print the logical vector
indicating duplicate values
dup_vector <- duplicated(iris)
print(dup_vector)
# 7. Extract duplicate elements from the dataset iris
duplicate_rows <- iris[dup_vector, ]
print(duplicate_rows)
# 8. Extract unique elements from the dataset iris
unique_rows <- iris[!dup_vector, ]
print(unique_rows)
# 9. Print the indices of duplicate elements in the dataset iris
duplicate_indices <- which(dup_vector)
print(duplicate_indices)
# 10. Print the indices of unique elements in the dataset iris
unique_indices <- which(!dup_vector)
print(unique_indices)
# 11. How many unique elements are in the dataset iris?
num_unique_elements <- length(unique_rows)
print(num_unique_elements)
# 12. How many duplicate elements are in the dataset iris?
num_duplicate_elements <- length(duplicate_rows)
print(num_duplicate_elements)
```

R Console

```
> # 1. Print the dataset iris
> print(iris)
    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
1            5.1         3.5          1.4         0.2     setosa
2            4.9         3.0          1.4         0.2     setosa
3            4.7         3.2          1.3         0.2     setosa
4            4.6         3.1          1.5         0.2     setosa
5            5.0         3.6          1.4         0.2     setosa
6            5.4         3.9          1.7         0.4     setosa
7            4.6         3.4          1.4         0.3     setosa
8            5.0         3.4          1.5         0.2     setosa
9            4.4         2.9          1.4         0.2     setosa
10           4.9         3.1          1.5         0.1     setosa
11           5.4         3.7          1.5         0.2     setosa
12           4.8         3.4          1.6         0.2     setosa
13           4.8         3.0          1.4         0.1     setosa
14           4.3         3.0          1.1         0.1     setosa
15           5.8         4.0          1.2         0.2     setosa
16           5.7         4.4          1.5         0.4     setosa
17           5.4         3.9          1.3         0.4     setosa
18           5.1         3.5          1.4         0.3     setosa
19           5.7         3.8          1.7         0.3     setosa
20           5.1         3.8          1.5         0.3     setosa
21           5.4         3.4          1.7         0.2     setosa
22           5.1         3.7          1.5         0.4     setosa
23           4.6         3.6          1.0         0.2     setosa
24           5.1         3.3          1.7         0.5     setosa
25           4.8         3.4          1.9         0.2     setosa
26           5.0         3.0          1.6         0.2     setosa
27           5.0         3.4          1.6         0.4     setosa
28           5.2         3.5          1.5         0.2     setosa
29           5.2         3.4          1.4         0.2     setosa
30           4.7         3.2          1.6         0.2     setosa
31           4.8         3.1          1.6         0.2     setosa
32           5.4         3.4          1.5         0.4     setosa
33           5.2         4.1          1.5         0.1     setosa
34           5.5         4.2          1.4         0.2     setosa
35           4.9         3.1          1.5         0.2     setosa
36           5.0         3.2          1.2         0.2     setosa
37           5.5         3.5          1.3         0.2     setosa
38           4.9         3.6          1.4         0.1     setosa
39           4.4         3.0          1.3         0.2     setosa
40           5.1         3.4          1.5         0.2     setosa
41           5.0         3.5          1.3         0.3     setosa
42           4.5         2.3          1.3         0.3     setosa
43           4.4         3.2          1.3         0.2     setosa
44           5.0         3.5          1.6         0.6     setosa
45           5.1         3.8          1.9         0.4     setosa
46           4.8         3.0          1.4         0.3     setosa
47           5.1         3.8          1.6         0.2     setosa
48           4.6         3.2          1.4         0.2     setosa
49           5.3         3.7          1.5         0.2     setosa
50           5.0         3.3          1.4         0.2     setosa
51           7.0         3.2          4.7         1.4 versicolor
52           6.4         3.2          4.5         1.5 versicolor
53           6.9         3.1          4.9         1.5 versicolor
54           5.5         2.3          4.0         1.3 versicolor
55           6.5         2.8          4.6         1.5 versicolor
56           5.7         2.8          4.5         1.3 versicolor
57           6.3         3.3          4.7         1.6 versicolor
58           4.9         2.4          3.3         1.0 versicolor
59           6.6         2.9          4.6         1.3 versicolor
60           5.2         2.7          3.9         1.4 versicolor
61           5.0         2.0          3.5         1.0 versicolor
62           5.9         3.0          4.2         1.5 versicolor
63           6.0         2.2          4.0         1.0 versicolor
64           6.1         2.9          4.7         1.4 versicolor
65           5.6         2.9          3.6         1.3 versicolor
```

66	6.7	3.1	4.4	1.4	versicolor
67	5.6	3.0	4.5	1.5	versicolor
68	5.8	2.7	4.1	1.0	versicolor
69	6.2	2.2	4.5	1.5	versicolor
70	5.6	2.5	3.9	1.1	versicolor
71	5.9	3.2	4.8	1.8	versicolor
72	6.1	2.8	4.0	1.3	versicolor
73	6.3	2.5	4.9	1.5	versicolor
74	6.1	2.8	4.7	1.2	versicolor
75	6.4	2.9	4.3	1.3	versicolor
76	6.6	3.0	4.4	1.4	versicolor
77	6.8	2.8	4.8	1.4	versicolor
78	6.7	3.0	5.0	1.7	versicolor
79	6.0	2.9	4.5	1.5	versicolor
80	5.7	2.6	3.5	1.0	versicolor
81	5.5	2.4	3.8	1.1	versicolor
82	5.5	2.4	3.7	1.0	versicolor
83	5.8	2.7	3.9	1.2	versicolor
84	6.0	2.7	5.1	1.6	versicolor
85	5.4	3.0	4.5	1.5	versicolor
86	6.0	3.4	4.5	1.6	versicolor
87	6.7	3.1	4.7	1.5	versicolor
88	6.3	2.3	4.4	1.3	versicolor
89	5.6	3.0	4.1	1.3	versicolor
90	5.5	2.5	4.0	1.3	versicolor
91	5.5	2.6	4.4	1.2	versicolor
92	6.1	3.0	4.6	1.4	versicolor
93	5.8	2.6	4.0	1.2	versicolor
94	5.0	2.3	3.3	1.0	versicolor
95	5.6	2.7	4.2	1.3	versicolor
96	5.7	3.0	4.2	1.2	versicolor
97	5.7	2.9	4.2	1.3	versicolor
98	6.2	2.9	4.3	1.3	versicolor
99	5.1	2.5	3.0	1.1	versicolor
100	5.7	2.8	4.1	1.3	versicolor
101	6.3	3.3	6.0	2.5	virginica
102	5.8	2.7	5.1	1.9	virginica
103	7.1	3.0	5.9	2.1	virginica
104	6.3	2.9	5.6	1.8	virginica
105	6.5	3.0	5.8	2.2	virginica
106	7.6	3.0	6.6	2.1	virginica
107	4.9	2.5	4.5	1.7	virginica
108	7.3	2.9	6.3	1.8	virginica
109	6.7	2.5	5.8	1.8	virginica
110	7.2	3.6	6.1	2.5	virginica
111	6.5	3.2	5.1	2.0	virginica
112	6.4	2.7	5.3	1.9	virginica
113	6.8	3.0	5.5	2.1	virginica
114	5.7	2.5	5.0	2.0	virginica
115	5.8	2.8	5.1	2.4	virginica
116	6.4	3.2	5.3	2.3	virginica
117	6.5	3.0	5.5	1.8	virginica
118	7.7	3.8	6.7	2.2	virginica
119	7.7	2.6	6.9	2.3	virginica
120	6.0	2.2	5.0	1.5	virginica
121	6.9	3.2	5.7	2.3	virginica
122	5.6	2.8	4.9	2.0	virginica
123	7.7	2.8	6.7	2.0	virginica
124	6.3	2.7	4.9	1.8	virginica
125	6.7	3.3	5.7	2.1	virginica
126	7.2	3.2	6.0	1.8	virginica
127	6.2	2.8	4.8	1.8	virginica
128	6.1	3.0	4.9	1.8	virginica
129	6.4	2.8	5.6	2.1	virginica
130	7.2	3.0	5.8	1.6	virginica
131	7.4	2.8	6.1	1.9	virginica
132	7.9	3.8	6.4	2.0	virginica
133	6.4	2.8	5.6	2.2	virginica
134	6.3	2.8	5.1	1.5	virginica
135	6.1	2.6	5.6	1.4	virginica

```
136          7.7         3.0          6.1         2.3  virginica
137          6.3         3.4          5.6         2.4  virginica
138          6.4         3.1          5.5         1.8  virginica
139          6.0         3.0          4.8         1.8  virginica
140          6.9         3.1          5.4         2.1  virginica
141          6.7         3.1          5.6         2.4  virginica
142          6.9         3.1          5.1         2.3  virginica
143          5.8         2.7          5.1         1.9  virginica
144          6.8         3.2          5.9         2.3  virginica
145          6.7         3.3          5.7         2.5  virginica
146          6.7         3.0          5.2         2.3  virginica
147          6.3         2.5          5.0         1.9  virginica
148          6.5         3.0          5.2         2.0  virginica
149          6.2         3.4          5.4         2.3  virginica
150          5.9         3.0          5.1         1.8  virginica
>
> # 2. Print the structure of the dataset iris
> str(iris)
'data.frame':   150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
>

>
> # 3. Print the summary of all the variables of the dataset iris
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
>
> # 4. How many variables (columns) are in the dataset iris?
> num_variables <- ncol(iris)
> print(num_variables)
[1] 5
>
> # 5. How many observations (rows) are in the dataset iris?
> num_observations <- nrow(iris)
> print(num_observations)
[1] 150
>
> # 6. Use duplicated() function to print the logical vector indicating duplicate values
> dup_vector <- duplicated(iris)
> print(dup_vector)
  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [19] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [55] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [91] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[109] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[127] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[145] FALSE FALSE FALSE FALSE FALSE FALSE
>
> # 7. Extract duplicate elements from the dataset iris
> duplicate_rows <- iris[dup_vector, ]
> print(duplicate_rows)
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
143          5.8         2.7          5.1         1.9 virginica
>
> # 8. Extract unique elements from the dataset iris
> unique_rows <- iris[!dup_vector, ]
> print(unique_rows)

    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
143          5.8         2.7          5.1         1.9 virginica
>
> # 8. Extract unique elements from the dataset iris
> unique_rows <- iris[!dup_vector, ]
> print(unique_rows)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
16	5.7	4.4	1.5	0.4	setosa
17	5.4	3.9	1.3	0.4	setosa
18	5.1	3.5	1.4	0.3	setosa
19	5.7	3.8	1.7	0.3	setosa
20	5.1	3.8	1.5	0.3	setosa
21	5.4	3.4	1.7	0.2	setosa
22	5.1	3.7	1.5	0.4	setosa
23	4.6	3.6	1.0	0.2	setosa
24	5.1	3.3	1.7	0.5	setosa
25	4.8	3.4	1.9	0.2	setosa
26	5.0	3.0	1.6	0.2	setosa
27	5.0	3.4	1.6	0.4	setosa
28	5.2	3.5	1.5	0.2	setosa
29	5.2	3.4	1.4	0.2	setosa
30	4.7	3.2	1.6	0.2	setosa
31	4.8	3.1	1.6	0.2	setosa
32	5.4	3.4	1.5	0.4	setosa
33	5.2	4.1	1.5	0.1	setosa
34	5.5	4.2	1.4	0.2	setosa
35	4.9	3.1	1.5	0.2	setosa
36	5.0	3.2	1.2	0.2	setosa
37	5.5	3.5	1.3	0.2	setosa
38	4.9	3.6	1.4	0.1	setosa
39	4.4	3.0	1.3	0.2	setosa
40	5.1	3.4	1.5	0.2	setosa
41	5.0	3.5	1.3	0.3	setosa
42	4.5	2.3	1.3	0.3	setosa
43	4.4	3.2	1.3	0.2	setosa
44	5.0	3.5	1.6	0.6	setosa
45	5.1	3.8	1.9	0.4	setosa
46	4.8	3.0	1.4	0.3	setosa
47	5.1	3.8	1.6	0.2	setosa
48	4.6	3.2	1.4	0.2	setosa
49	5.3	3.7	1.5	0.2	setosa
50	5.0	3.3	1.4	0.2	setosa
51	7.0	3.2	4.7	1.4	versicolor
52	6.4	3.2	4.5	1.5	versicolor
53	6.9	3.1	4.9	1.5	versicolor
54	5.5	2.3	4.0	1.3	versicolor
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
58	4.9	2.4	3.3	1.0	versicolor
59	6.6	2.9	4.6	1.3	versicolor
60	5.2	2.7	3.9	1.4	versicolor
61	5.0	2.0	3.5	1.0	versicolor
62	5.9	3.0	4.2	1.5	versicolor
63	6.0	2.2	4.0	1.0	versicolor
64	6.1	2.9	4.7	1.4	versicolor
65	5.6	2.9	3.6	1.3	versicolor

66	6.7	3.1	4.4	1.4	versicolor
67	5.6	3.0	4.5	1.5	versicolor
68	5.8	2.7	4.1	1.0	versicolor
69	6.2	2.2	4.5	1.5	versicolor
70	5.6	2.5	3.9	1.1	versicolor
71	5.9	3.2	4.8	1.8	versicolor
72	6.1	2.8	4.0	1.3	versicolor
73	6.3	2.5	4.9	1.5	versicolor
74	6.1	2.8	4.7	1.2	versicolor
75	6.4	2.9	4.3	1.3	versicolor
76	6.6	3.0	4.4	1.4	versicolor
77	6.8	2.8	4.8	1.4	versicolor
78	6.7	3.0	5.0	1.7	versicolor
79	6.0	2.9	4.5	1.5	versicolor
80	5.7	2.6	3.5	1.0	versicolor
81	5.5	2.4	3.8	1.1	versicolor
82	5.5	2.4	3.7	1.0	versicolor
83	5.8	2.7	3.9	1.2	versicolor
84	6.0	2.7	5.1	1.6	versicolor
85	5.4	3.0	4.5	1.5	versicolor
86	6.0	3.4	4.5	1.6	versicolor
87	6.7	3.1	4.7	1.5	versicolor
88	6.3	2.3	4.4	1.3	versicolor
89	5.6	3.0	4.1	1.3	versicolor
90	5.5	2.5	4.0	1.3	versicolor
91	5.5	2.6	4.4	1.2	versicolor
92	6.1	3.0	4.6	1.4	versicolor
93	5.8	2.6	4.0	1.2	versicolor
94	5.0	2.3	3.3	1.0	versicolor
95	5.6	2.7	4.2	1.3	versicolor
96	5.7	3.0	4.2	1.2	versicolor
97	5.7	2.9	4.2	1.3	versicolor
98	6.2	2.9	4.3	1.3	versicolor
99	5.1	2.5	3.0	1.1	versicolor
100	5.7	2.8	4.1	1.3	versicolor
101	6.3	3.3	6.0	2.5	virginica
102	5.8	2.7	5.1	1.9	virginica
103	7.1	3.0	5.9	2.1	virginica
104	6.3	2.9	5.6	1.8	virginica
105	6.5	3.0	5.8	2.2	virginica
106	7.6	3.0	6.6	2.1	virginica
107	4.9	2.5	4.5	1.7	virginica
108	7.3	2.9	6.3	1.8	virginica
109	6.7	2.5	5.8	1.8	virginica
110	7.2	3.6	6.1	2.5	virginica
111	6.5	3.2	5.1	2.0	virginica
112	6.4	2.7	5.3	1.9	virginica
113	6.8	3.0	5.5	2.1	virginica
114	5.7	2.5	5.0	2.0	virginica
115	5.8	2.8	5.1	2.4	virginica
116	6.4	3.2	5.3	2.3	virginica
117	6.5	3.0	5.5	1.8	virginica
118	7.7	3.8	6.7	2.2	virginica
119	7.7	2.6	6.9	2.3	virginica
120	6.0	2.2	5.0	1.5	virginica
121	6.9	3.2	5.7	2.3	virginica
122	5.6	2.8	4.9	2.0	virginica
123	7.7	2.8	6.7	2.0	virginica
124	6.3	2.7	4.9	1.8	virginica
125	6.7	3.3	5.7	2.1	virginica
126	7.2	3.2	6.0	1.8	virginica
127	6.2	2.8	4.8	1.8	virginica
128	6.1	3.0	4.9	1.8	virginica
129	6.4	2.8	5.6	2.1	virginica
130	7.2	3.0	5.8	1.6	virginica
131	7.4	2.8	6.1	1.9	virginica
132	7.9	3.8	6.4	2.0	virginica
133	6.4	2.8	5.6	2.2	virginica
134	6.3	2.8	5.1	1.5	virginica
135	6.1	2.6	5.6	1.4	virginica

```
136          7.7        3.0         6.1        2.3  virginica
137          6.3        3.4         5.6        2.4  virginica
138          6.4        3.1         5.5        1.8  virginica
139          6.0        3.0         4.8        1.8  virginica
140          6.9        3.1         5.4        2.1  virginica
141          6.7        3.1         5.6        2.4  virginica
142          6.9        3.1         5.1        2.3  virginica
144          6.8        3.2         5.9        2.3  virginica
145          6.7        3.3         5.7        2.5  virginica
146          6.7        3.0         5.2        2.3  virginica
147          6.3        2.5         5.0        1.9  virginica
148          6.5        3.0         5.2        2.0  virginica
149          6.2        3.4         5.4        2.3  virginica
150          5.9        3.0         5.1        1.8  virginica
>
> # 9. Print the indices of duplicate elements in the dataset iris
> duplicate_indices <- which(dup_vector)
> print(duplicate_indices)
[1] 143
>
> # 10. Print the indices of unique elements in the dataset iris
> unique_indices <- which(!dup_vector)
> print(unique_indices)
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27
 [28]  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81
 [82]  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
[109] 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135
[136] 136 137 138 139 140 141 142 144 145 146 147 148 149 150
>
> # 11. How many unique elements are in the dataset iris?
> num_unique_elements <- length(unique_rows)
> print(num_unique_elements)
[1] 5
>
> # 12. How many duplicate elements are in the dataset iris?
> num_duplicate_elements <- length(duplicate_rows)
> print(num_duplicate_elements)
[1] 5
> |
```

Q.4 Consider a vector x <-c(1,5,9,67,NA,32,NA,NA,12)

1] Check for missing values using is.na() and completecases()

2] Count missing values as sum, colSums functions

3] Handling missing values using omit, replace with

0 and replace with previous element and mean, median, Sum values

```
# Define the vector
x <- c(1, 5, 9, 67, NA, 32, NA, NA, 12)
x
# 1. Check for missing values using is.na() and complete.cases()
missing_values_isna <- is.na(x)
missing_values_completecases <- !complete.cases(x)
print(missing_values_isna)
print(missing_values_completecases)
# 2. Count missing values using sum() and colSums() functions
count_missing_values_sum <- sum(is.na(x))
count_missing_values_colsums <- colSums(is.na(x))
print(count_missing_values_sum)
print(count_missing_values_colsums)
```

```
# 3. Handling missing values
# Option 1: Omit missing values
x_omitted <- na.omit(x)
print(x_omitted)
# Option 2: Replace missing values with 0
x_zero_filled <- replace(x, is.na(x), 0)
print(x_zero_filled)
# Option 3: Replace missing values with previous element
x_previous_filled <- zoo::na.locf(x)
print(x_previous_filled)
# Option 4: Replace missing values with mean
x_mean_filled <- replace(x, is.na(x), mean(x, na.rm = TRUE))
print(x_mean_filled)
# Option 5: Replace missing values with median
x_median_filled <- replace(x, is.na(x), median(x, na.rm = TRUE))
print(x_median_filled)
# Option 6: Replace missing values with sum
x_sum_filled <- replace(x, is.na(x), sum(x, na.rm = TRUE))
print(x_sum_filled)
```

R Console

```
> # Define the vector
> x <- c(1, 5, 9, 67, NA, 32, NA, NA, 12)
> x
[1]  1  5  9 67 NA 32 NA NA 12
> # 1. Check for missing values using is.na() and complete.cases()
> missing_values_isna <- is.na(x)
> missing_values_completecases <- !complete.cases(x)
> print(missing_values_isna)
[1] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE
> print(missing_values_completecases)
[1] FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE
> # 2. Count missing values using sum() and colSums() functions
> count_missing_values_sum <- sum(is.na(x))
> count_missing_values_colsums <- colSums(is.na(x))
Error in colSums(is.na(x)) :
  'x' must be an array of at least two dimensions
> print(count_missing_values_sum)
[1] 3
> print(count_missing_values_colsums)
Error in print(count_missing_values_colsums) :
  object 'count_missing_values_colsums' not found
> # 3. Handling missing values
> # Option 1: Omit missing values
> x_omitted <- na.omit(x)
> print(x_omitted)
[1]  1  5  9 67 32 12
attr(,"na.action")
```

```
[1] 5 7 8
attr(,"class")
[1] "omit"
> # Option 2: Replace missing values with 0
> x_zero_filled <- replace(x, is.na(x), 0)
> print(x_zero_filled)
[1]  1  5  9 67  0 32  0  0 12
> # Option 3: Replace missing values with previous element
> x_previous_filled <- zoo::na.locf(x)
Error in loadNamespace(name) : there is no package called 'zoo'
> print(x_previous_filled)
Error in print(x_previous_filled) : object 'x_previous_filled' not found
> # Option 4: Replace missing values with mean
> x_mean_filled <- replace(x, is.na(x), mean(x, na.rm = TRUE))
> print(x_mean_filled)
[1]  1  5  9 67 21 32 21 21 12
> # Option 5: Replace missing values with median
> x_median_filled <- replace(x, is.na(x), median(x, na.rm = TRUE))
> print(x_median_filled)
[1]  1.0  5.0  9.0 67.0 10.5 32.0 10.5 10.5 12.0
> # Option 6: Replace missing values with sum
> x_sum_filled <- replace(x, is.na(x), sum(x, na.rm = TRUE))
> print(x_sum_filled)
[1]   1   5   9  67 126  32 126 126  12
> |
```

Q.5 Create a dataframe with missing values
df <- data.frame(
ID = c(1, 2, 3, NA, 5),
Name = c("John", "Alice", NA, "Bob", "Jane")
)
1] Check for missing values using is.na( ) and completecases()
2] Count missing values as sum, colSums functions
3] Handling missing values using omit, replace with 0 and replace with previous element and mean, median, Sum values.

```
# Create the dataframe with missing values
df <- data.frame(
  ID = c(1, 2, 3, NA, 5),
  Name = c("John", "Alice", NA, "Bob", "Jane")
)
df
# 1. Check for missing values using is.na() and complete.cases()
missing_values_isna <- is.na(df)
missing_values_completecases <- !complete.cases(df)
print(missing_values_isna)
```

```
print(missing_values_completecases)

# 2. Count missing values using sum() and colSums() functions
count_missing_values_sum <- sum(is.na(df))
count_missing_values_colsums <- colSums(is.na(df))
print(count_missing_values_sum)
print(count_missing_values_colsums)

# 3. Handling missing values
# Option 1: Omit rows with missing values
df_omitted <- na.omit(df)
print(df_omitted)
# Option 2: Replace missing values with 0
df_zero_filled <- replace(df, is.na(df), 0)
print(df_zero_filled)
# Option 3: Replace missing values with previous element
df_previous_filled <- zoo::na.locf(df)
print(df_previous_filled)
# Option 4: Replace missing values with mean
df_mean_filled <- replace(df, is.na(df), mean(df, na.rm = TRUE))
print(df_mean_filled)
# Option 5: Replace missing values with median
df_median_filled <- replace(df, is.na(df), median(df, na.rm =
TRUE))
print(df_median_filled)
# Option 6: Replace missing values with sum
df_sum_filled <- replace(df, is.na(df), sum(df, na.rm = TRUE))
df_sum_filled
```

R Console

```
> # Create the dataframe with missing values
> df <- data.frame(
+   ID = c(1, 2, 3, NA, 5),
+   Name = c("John", "Alice", NA, "Bob", "Jane")
+ )
> df
  ID  Name
1  1  John
2  2 Alice
3  3  <NA>
4 NA   Bob
5  5  Jane
> # 1. Check for missing values using is.na() and complete.cases()
> missing_values_isna <- is.na(df)
> missing_values_completecases <- !complete.cases(df)
> print(missing_values_isna)
        ID  Name
[1,] FALSE FALSE
[2,] FALSE FALSE
[3,] FALSE  TRUE
[4,]  TRUE FALSE
[5,] FALSE FALSE
> print(missing_values_completecases)
[1] FALSE FALSE  TRUE  TRUE FALSE
>
> # 2. Count missing values using sum() and colSums() functions
> count_missing_values_sum <- sum(is.na(df))
> count_missing_values_colsums <- colSums(is.na(df))
> print(count_missing_values_sum)
[1] 2
> print(count_missing_values_colsums)
  ID Name
   1    1
>
> # 3. Handling missing values
> # Option 1: Omit rows with missing values
> df_omitted <- na.omit(df)
> print(df_omitted)
  ID  Name
1  1  John
2  2 Alice
5  5  Jane
> # Option 2: Replace missing values with 0
> df_zero_filled <- replace(df, is.na(df), 0)
> print(df_zero_filled)
  ID  Name
1  1  John
2  2 Alice
3  3     0
4  0   Bob
5  5  Jane
> # Option 3: Replace missing values with previous element
> df_previous_filled <- zoo::na.locf(df)
Error in loadNamespace(name) : there is no package called 'zoo'
```

```
> df_previous_filled <- zoo::na.locf(df)
Error in loadNamespace(name) : there is no package called 'zoo'
> print(df_previous_filled)
Error in print(df_previous_filled) :
  object 'df_previous_filled' not found
> # Option 4: Replace missing values with mean
> df_mean_filled <- replace(df, is.na(df), mean(df, na.rm = TRUE))
Warning message:
In mean.default(df, na.rm = TRUE) :
  argument is not numeric or logical: returning NA
> print(df_mean_filled)
  ID  Name
1  1  John
2  2 Alice
3  3  <NA>
4 NA   Bob
5  5  Jane
> # Option 5: Replace missing values with median
> df_median_filled <- replace(df, is.na(df), median(df, na.rm = TRUE))
Error in median.default(df, na.rm = TRUE) : need numeric data
> print(df_median_filled)
Error in print(df_median_filled) : object 'df_median_filled' not found
> # Option 6: Replace missing values with sum
> df_sum_filled <- replace(df, is.na(df), sum(df, na.rm = TRUE))
Error in FUN(X[[i]], ...) :
  only defined on a data frame with all numeric variables
> df_sum_filled
```

Q.6 Consider a 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] Use the function is.na() to find whether any missing values are in the dataset airquality
7] Print the indices of the missing values in the dataset airquality in column major representation
8] Print the indices of the missing values in the dataset airquality in row major representation
9] Print indices of the missing values in row and column numberwise (Hint: Use function which() and argument arr.ind = TRUE)
10] How many missing values are in the dataset airquality?

11] Which variables are the missing values concentrated in?
12] How would you omit all rows containing missing values?
13] Print the records without missing values in the dataset airquality using the function complete.cases()
14] Print the records without missing values in the dataset airquality using the function na.omit()
15] Print the records without missing values in the dataset airquality using the function na.exclude()
16] Print the records containing missing values in the dataset airquality using the function complete.cases()

```
# 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. Use the function is.na() to find whether any missing values
are in the dataset airquality
has_missing_values <- any(is.na(airquality))
print(has_missing_values)

# 7. Print the indices of the missing values in the dataset
airquality in column major representation
```

```
missing_values_indices_col <- which(is.na(airquality), arr.ind =
TRUE)
print(missing_values_indices_col)

# 8. Print the indices of the missing values in the dataset
airquality in row major representation
missing_values_indices_row <- which(t(is.na(airquality)), arr.ind
= TRUE)
print(missing_values_indices_row)

# 9. Print indices of the missing values in row and column
numberwise
missing_values_indices <- which(is.na(airquality), arr.ind =
TRUE)
print(missing_values_indices)

# 10. How many missing values are in the dataset airquality
num_missing_values <- sum(is.na(airquality))
print(num_missing_values)

# 11. Which variables are the missing values concentrated in
variables_with_missing_values <-
colnames(airquality)[colSums(is.na(airquality)) > 0]
print(variables_with_missing_values)

# 12. Omit all rows containing missing values
airquality_no_missing <- na.omit(airquality)
print(airquality_no_missing)

# 13. Print the records without missing values using
complete.cases()
complete_cases <- airquality[complete.cases(airquality), ]
print(complete_cases)

# 14. Print the records without missing values using na.omit()
omit_records <- na.omit(airquality)
print(omit_records)
```

```
# 15. Print the records without missing values using na.exclude()
exclude_records <- na.exclude(airquality)
print(exclude_records)

# 16. Print the records containing missing values using
complete.cases()
missing_value_records <- airquality[!complete.cases(airquality),
]
print(missing_value_records)
```

```
> # 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. Use the function is.na() to find whether any missing values are in the dataset airquality
> has_missing_values <- any(is.na(airquality))
> print(has_missing_values)
[1] TRUE
```

```
>
> # 7. Print the indices of the missing values in the dataset airquality in column major representation
> missing_values_indices_col <- which(is.na(airquality), arr.ind = TRUE)
> print(missing_values_indices_col)
      row col
 [1,]   5   1
 [2,]  10   1
 [3,]  25   1
 [4,]  26   1
 [5,]  27   1
 [6,]  32   1
 [7,]  33   1
 [8,]  34   1
 [9,]  35   1
[10,]  36   1
[11,]  37   1
[12,]  39   1
[13,]  42   1
[14,]  43   1
[15,]  45   1
[16,]  46   1
[17,]  52   1
[18,]  53   1
[19,]  54   1
[20,]  55   1
[21,]  56   1
[22,]  57   1
[23,]  58   1
[24,]  59   1
[25,]  60   1
[26,]  61   1
[27,]  65   1
[28,]  72   1
[29,]  75   1
[30,]  83   1
[31,]  84   1
[32,] 102   1
[33,] 103   1
[34,] 107   1
[35,] 115   1
[36,] 119   1
[37,] 150   1
[38,]   5   2
[39,]   6   2
[40,]  11   2
[41,]  27   2
[42,]  96   2
[43,]  97   2
[44,]  98   2
>
> # 8. Print the indices of the missing values in the dataset airquality in row major representation
> missing_values_indices_row <- which(t(is.na(airquality)), arr.ind = TRUE)
> print(missing_values_indices_row)
        row col
Ozone     1   5
Solar.R   2   5
Solar.R   2   6
Ozone     1  10
Solar.R   2  11
Ozone     1  25
Ozone     1  26
Ozone     1  27
Solar.R   2  27
Ozone     1  32
Ozone     1  33
Ozone     1  34
Ozone     1  35
Ozone     1  36
Ozone     1  37
Ozone     1  39
Ozone     1  42
```

```
Ozone     1  43
Ozone     1  45
Ozone     1  46
Ozone     1  52
Ozone     1  53
Ozone     1  54
Ozone     1  55
Ozone     1  56
Ozone     1  57
Ozone     1  58
Ozone     1  59
Ozone     1  60
Ozone     1  61
Ozone     1  65
Ozone     1  72
Ozone     1  75
Ozone     1  83
Ozone     1  84
Solar.R   2  96
Solar.R   2  97
Solar.R   2  98
Ozone     1 102
Ozone     1 103
Ozone     1 107
Ozone     1 115
Ozone     1 119
Ozone     1 150
>
> # 9. Print indices of the missing values in row and column numberwise
> missing_values_indices <- which(is.na(airquality), arr.ind = TRUE)
> print(missing_values_indices)
      row col
 [1,]   5   1
 [2,]  10   1
 [3,]  25   1
 [4,]  26   1
 [5,]  27   1
 [6,]  32   1
 [7,]  33   1
 [8,]  34   1
 [9,]  35   1
[10,]  36   1
[11,]  37   1
[12,]  39   1
[13,]  42   1
[14,]  43   1
[15,]  45   1
[16,]  46   1
[17,]  52   1
[18,]  53   1
[19,]  54   1
[20,]  55   1
[21,]  56   1
[22,]  57   1
[23,]  58   1
[24,]  59   1
[25,]  60   1
[26,]  61   1
[27,]  65   1
[28,]  72   1
[29,]  75   1
[30,]  83   1
[31,]  84   1
[32,] 102   1
[33,] 103   1
[34,] 107   1
[35,] 115   1
[36,] 119   1
[37,] 150   1
[38,]   5   2
[39,]   6   2
```

```
[40,]  11   2
[41,]  27   2
[42,]  96   2
[43,]  97   2
[44,]  98   2
>
> # 10. How many missing values are in the dataset airquality
> num_missing_values <- sum(is.na(airquality))
> print(num_missing_values)
[1] 44
>
> # 11. Which variables are the missing values concentrated in
> variables_with_missing_values <- colnames(airquality)[colSums(is.na(airquality)) > 0]
> print(variables_with_missing_values)
[1] "Ozone"   "Solar.R"
>
> # 12. Omit all rows containing missing values
> airquality_no_missing <- na.omit(airquality)
> print(airquality_no_missing)
    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
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
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
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
38     29     127  9.7   82     6   7
40     71     291 13.8   90     6   9
41     39     323 11.5   87     6  10
44     23     148  8.0   82     6  13
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
62    135     269  4.1   84     7   1
63     49     248  9.2   85     7   2
64     32     236  9.2   81     7   3
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
73     10     264 14.3   73     7  12
74     27     175 14.9   81     7  13
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
```

```
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
99    122     255  4.0   89     8   7
100    89     229 10.3   90     8   8
101   110     207  8.0   90     8   9
104    44     192 11.5   86     8  12
105    28     273 11.5   82     8  13
106    65     157  9.7   80     8  14
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
116    45     212  9.7   79     8  24
117   168     238  3.4   81     8  25
118    73     215  8.0   86     8  26
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
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
>
> # 13. Print the records without missing values using complete.cases()
> complete_cases <- airquality[complete.cases(airquality), ]
> print(complete_cases)
    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
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
```

9	8	19	20.1	61	5	9
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
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
38	29	127	9.7	82	6	7
40	71	291	13.8	90	6	9
41	39	323	11.5	87	6	10
44	23	148	8.0	82	6	13
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
62	135	269	4.1	84	7	1
63	49	248	9.2	85	7	2
64	32	236	9.2	81	7	3
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
73	10	264	14.3	73	7	12
74	27	175	14.9	81	7	13
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
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
99	122	255	4.0	89	8	7
100	89	229	10.3	90	8	8
101	110	207	8.0	90	8	9
104	44	192	11.5	86	8	12
105	28	273	11.5	82	8	13
106	65	157	9.7	80	8	14
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
116	45	212	9.7	79	8	24
117	168	238	3.4	81	8	25

```
118    73     215  8.0   86     8  26
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
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
>
> # 14. Print the records without missing values using na.omit()
> omit_records <- na.omit(airquality)
> print(omit_records)
    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
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
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
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
38     29     127  9.7   82     6   7
40     71     291 13.8   90     6   9
41     39     323 11.5   87     6  10
44     23     148  8.0   82     6  13
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
62	135	269	4.1	84	7	1
63	49	248	9.2	85	7	2
64	32	236	9.2	81	7	3
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
73	10	264	14.3	73	7	12
74	27	175	14.9	81	7	13
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
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
99	122	255	4.0	89	8	7
100	89	229	10.3	90	8	8
101	110	207	8.0	90	8	9
104	44	192	11.5	86	8	12
105	28	273	11.5	82	8	13
106	65	157	9.7	80	8	14
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
116	45	212	9.7	79	8	24
117	168	238	3.4	81	8	25
118	73	215	8.0	86	8	26
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
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. Print the records without missing values using na.exclude()
> exclude_records <- na.exclude(airquality)
> print(exclude_records)
    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
7      23     299  8.6   65     5   7
8      19      99 13.8   59     5   8
9       8      19 20.1   61     5   9
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
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
38     29     127  9.7   82     6   7
40     71     291 13.8   90     6   9
41     39     323 11.5   87     6  10
44     23     148  8.0   82     6  13
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
62    135     269  4.1   84     7   1
63     49     248  9.2   85     7   2
64     32     236  9.2   81     7   3
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
73     10     264 14.3   73     7  12
74     27     175 14.9   81     7  13
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
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
99    122     255  4.0   89     8   7
100    89     229 10.3   90     8   8
101   110     207  8.0   90     8   9
104    44     192 11.5   86     8  12
105    28     273 11.5   82     8  13
106    65     157  9.7   80     8  14
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
116    45     212  9.7   79     8  24
117   168     238  3.4   81     8  25
118    73     215  8.0   86     8  26
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
151    14     191 14.3   75     9  28
152    18     131  8.0   76     9  29
153    20     223 11.5   68     9  30
>
> # 16. Print the records containing missing values using complete.cases()
> missing_value_records <- airquality[!complete.cases(airquality), ]
> print(missing_value_records)
    Ozone Solar.R Wind Temp Month Day
5      NA      NA 14.3   56     5   5
6      28      NA 14.9   66     5   6
10     NA     194  8.6   69     5  10
11      7      NA  6.9   74     5  11
25     NA      66 16.6   57     5  25
26     NA     266 14.9   58     5  26
27     NA      NA  8.0   57     5  27
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
```

39	NA	273	6.9	87	6	8
42	NA	259	10.9	93	6	11
43	NA	250	9.2	92	6	12
45	NA	332	13.8	80	6	14
46	NA	322	11.5	79	6	15
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
65	NA	101	10.9	84	7	4
72	NA	139	8.6	82	7	11
75	NA	291	14.9	91	7	14
83	NA	258	9.7	81	7	22
84	NA	295	11.5	82	7	23
96	78	NA	6.9	86	8	4
97	35	NA	7.4	85	8	5
98	66	NA	4.6	87	8	6
102	NA	222	8.6	92	8	10
103	NA	137	11.5	86	8	11
107	NA	64	11.5	79	8	15
115	NA	255	12.6	75	8	23
119	NA	153	5.7	88	8	27
150	NA	145	13.2	77	9	27

Screenshots were too large to be pasted so I pasted the output text from R console.