

LAB

STATISTICS WITH R PROGRAMMING FOR VISUALIZATION

COURSE CODE: ITA0435

NAME: P. NIKITHA

REG NO: 192324226

1. Create numeric, character, and logical vectors and display type and content.

CODE:

```
num_vec <- c(10, 20, 30)

char_vec <- c("A", "B", "C")

log_vec <- c(TRUE, FALSE, TRUE)

print(num_vec)

print(char_vec)

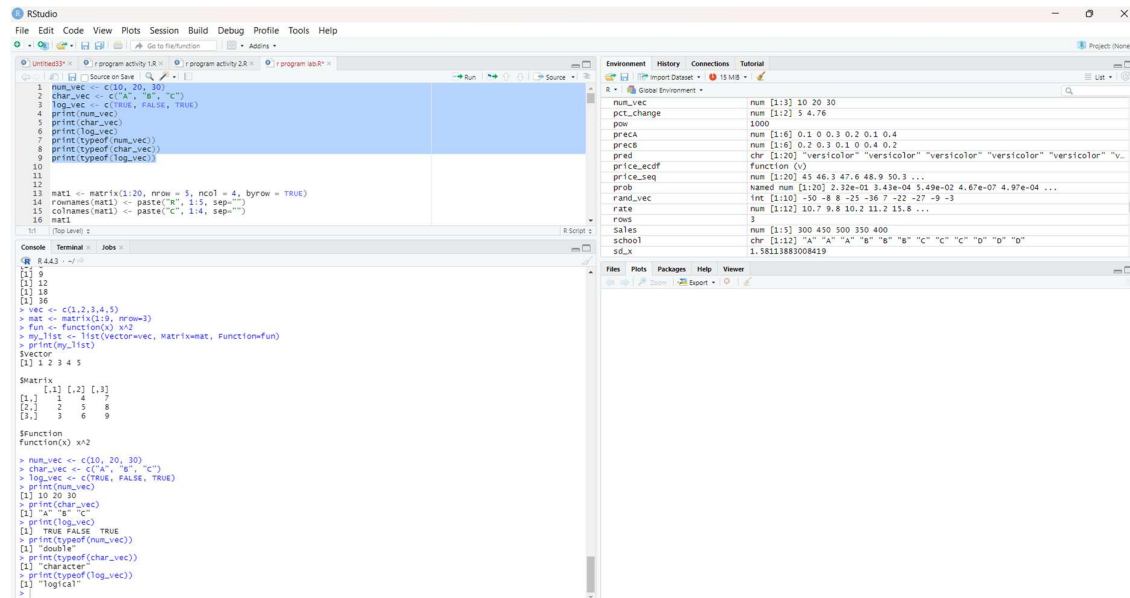
print(log_vec)

print(typeof(num_vec))

print(typeof(char_vec))

print(typeof(log_vec))
```

OUTPUT:



2. Create labeled matrices (5×4, 3×3, 2×2) filled by row/column.

CODE:

```
mat1 <- matrix(1:20, nrow = 5, ncol = 4, byrow = TRUE)

rownames(mat1) <- paste("R", 1:5, sep="")

colnames(mat1) <- paste("C", 1:4, sep="")
```

mat1

```
mat2 <- matrix(1:9, nrow = 3, ncol = 3)
```

```
rownames(mat2) <- paste("R", 1:3, sep="")
```

```
colnames(mat2) <- paste("C", 1:3, sep="")
```

mat2

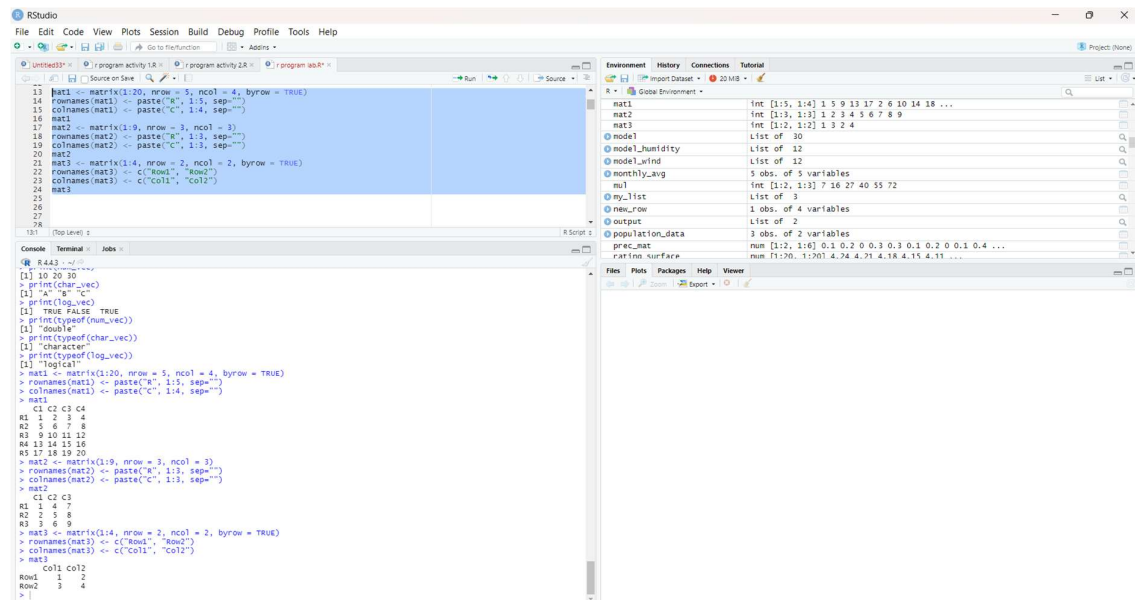
```
mat3 <- matrix(1:4, nrow = 2, ncol = 2, byrow = TRUE)
```

```
rownames(mat3) <- c("Row1", "Row2")
```

```
colnames(mat3) <- c("Col1", "Col2")
```

mat3

OUTPUT:



The screenshot shows the RStudio interface. The script editor on the left contains the following R code:

```
13 mat1 <- matrix(1:20, nrow = 5, ncol = 4, byrow = TRUE)
14 rownames(mat1) <- paste("R", 1:5, sep="")
15 colnames(mat1) <- paste("C", 1:4, sep="")
16 mat1
17 mat2 <- matrix(1:9, nrow = 3, ncol = 3)
18 rownames(mat2) <- paste("R", 1:3, sep="")
19 colnames(mat2) <- paste("C", 1:3, sep="")
20 mat2
21 mat3 <- matrix(1:4, nrow = 2, ncol = 2, byrow = TRUE)
22 rownames(mat3) <- c("Row1", "Row2")
23 colnames(mat3) <- c("Col1", "Col2")
24 mat3
```

The console on the bottom left shows the output of the code:

```
R 4.4.3 ~ /...
> print(char_vec)
[1] "a" "b" "c"
> print(log_vec)
[1] TRUE FALSE TRUE
> print(typeof(num_vec))
[1] "double"
> print(typeof(char_vec))
[1] "character"
> print(typeof(log_vec))
[1] "logical"
> mat1 <- matrix(1:20, nrow = 5, ncol = 4, byrow = TRUE)
> rownames(mat1) <- paste("R", 1:5, sep="")
> colnames(mat1) <- paste("C", 1:4, sep="")
> mat1
      C1 C2 C3 C4
R1    1  2  3  4
R2    5  6  7  8
R3    9 10 11 12
R4   13 14 15 16
R5   17 18 19 20
> mat2 <- matrix(1:9, nrow = 3, ncol = 3)
> rownames(mat2) <- paste("R", 1:3, sep="")
> colnames(mat2) <- paste("C", 1:3, sep="")
> mat2
      C1 C2 C3
R1    1  4  7
R2    2  5  8
R3    3  6  9
> mat3 <- matrix(1:4, nrow = 2, ncol = 2, byrow = TRUE)
> rownames(mat3) <- c("Row1", "Row2")
> colnames(mat3) <- c("Col1", "Col2")
> mat3
      Col1 Col2
Row1    1    2
Row2    3    4
```

3. Write an R program to create and display a 3D array with specified rows, columns, and tables.

CODE:

```
rows <- 3
```

```
cols <- 3
```

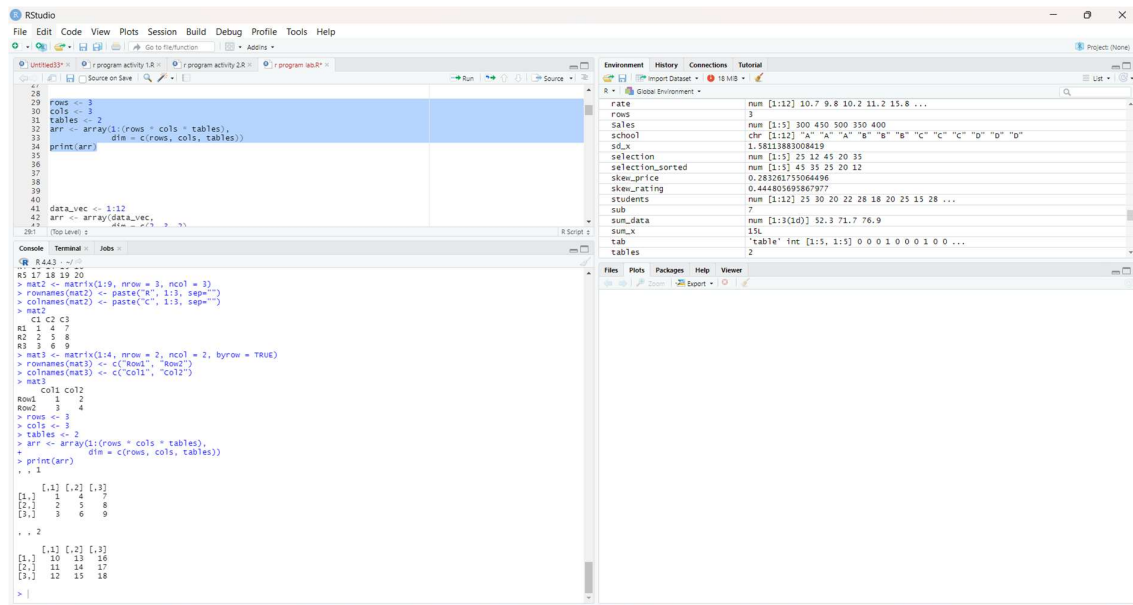
```
tables <- 2
```

```
arr <- array(1:(rows * cols * tables),
```

```
dim = c(rows, cols, tables))
```

```
print(arr)
```

OUTPUT:



4. Create arrays from vectors with dimension names, print specific elements.

CODE:

```
data_vec <- 1:12
```

```
arr <- array(data_vec,
```

```
  dim = c(2, 3, 2),
```

```
  dimnames = list(
```

```
    Row = c("R1", "R2"),
```

```
    Column = c("C1", "C2", "C3"),
```

```
    Table = c("T1", "T2")
```

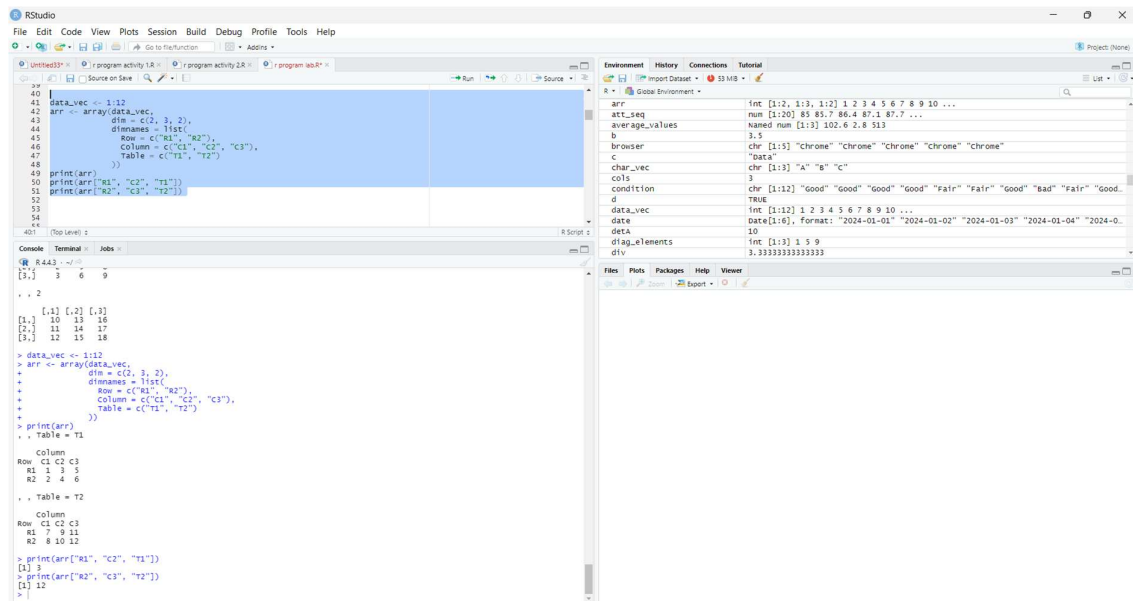
```
  ))
```

```
print(arr)
```

```
print(arr["R1", "C2", "T1"])
```

```
print(arr["R2", "C3", "T2"])
```

OUTPUT:



5. Create and manipulate factor variables (e.g., women's dataset heights, random LETTERS sample).

CODE:

```
data(women)
```

```
height_group <- factor(
  ifelse(women$height < 65, "Short",
         ifelse(women$height <= 70, "Medium", "Tall"))
)
```

```
print(height_group)
```

```
levels(height_group)
```

```
set.seed(1)
```

```
letters_sample <- sample(LETTERS[1:5], 10, replace = TRUE)
```

```
letters_factor <- factor(letters_sample)
```

```
print(letters_factor)
```

```
levels(letters_factor)
```

```
table(letters_factor)
```

OUTPUT:

```

57 [data:women]
58 height_group <- factor(
59   ifelse(women$height < 65, "short",
60         ifelse(women$height <= 70, "medium", "tall"))
61 )
62 print(height_group)
63 levels(height_group)
64 set.seed(1)
65 letters_sample <- sample(LETTERS[1:5], 10, replace = TRUE)
66 letters_factor <- factor(letters_sample)
67 print(letters_factor)
68 levels(letters_factor)
69 table(letters_factor)
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Environment

Object	Class	Attributes
women	data.frame	15 obs. of 2 variables
year_population_data	data.frame	5 obs. of 2 variables
values	num	[1:20] 22 23.5 24.9 26.4 27.9 ...
arr	int	[1:2, 1:3, 1:2] 1 2 3 4 5 6 7 8 9 10 ...
att_seq	num	[1:20] 85 85.7 86.4 87.1 87.7 ...
average_values	named num	[1:3] 102.6 2.8 53.3
b	5.5	
browser	chr	[1:5] "chrome" "chrome" "chrome" "chrome" "chrome"
c	"data"	
char_vec	chr	[1:1] "a" "b" "c"
cols	3	
condition	chr	[1:12] "Good" "Good" "Good" "Good" "Fair" "Good" "Bad" "Fair" "Good"
d	true	

6. Create an R list containing vectors, matrices, and functions; display contents.

CODE:

```
vec <- c(1,2,3,4,5)
```

```
mat <- matrix(1:9, nrow=3)
```

```
fun <- function(x) x^2
```

```
my_list <- list(Vector=vec, Matrix=mat, Function=fun)
```

```
print(my_list)
```

OUTPUT:

```

69 levels(letters_factor)
70 table(letters_factor)
71
72
73
74 vec <- c(1,2,3,4,5)
75 mat <- matrix(1:9, nrow=3)
76 fun <- function(x) x^2
77 my_list <- list(Vector=vec, Matrix=mat, Function=fun)
78 print(my_list)
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Environment

Object	Class	Attributes
values	num	[1:4] 4 9 16 25
vec	num	[1:5] 1 2 3 4 5
volume_seq	num	[1:20] 2.2 2.37 2.34 2.41 2.47 ...
wind_seq	num	[1:20] 12 12.4 12.8 13.3 13.7 ...
x	int	[1:5] 1 2 3 4 5
x_round	num	[1:20] -0.6 0.2 -0.8 1.6 0.3 -0.8 0.5 0.7 0.6 -0.3 ...
y	logi	[1:3] FALSE TRUE TRUE
year	num	[1:5] 2019 2020 2022 2023 2024
z	30	
add_analyze	function	function (a, b, c)
avg_salary	function	function (df)
even_count	function	function (x)
fact	function	function (n)
fun	function	function (x)

7. Write R programs for basic tasks: Factors of a number, generate a vector of 10 random integers between -50 and 50, print numbers 1–100 with FizzBuzz logic.

CODE:

```
rand_vec <- sample(-50:50, 10, replace = TRUE)

print(rand_vec)

for(i in 1:100){

  if(i %% 3 == 0 && i %% 5 == 0){

    print("FizzBuzz")

  } else if(i %% 3 == 0){

    print("Fizz")

  } else if(i %% 5 == 0){

    print("Buzz")

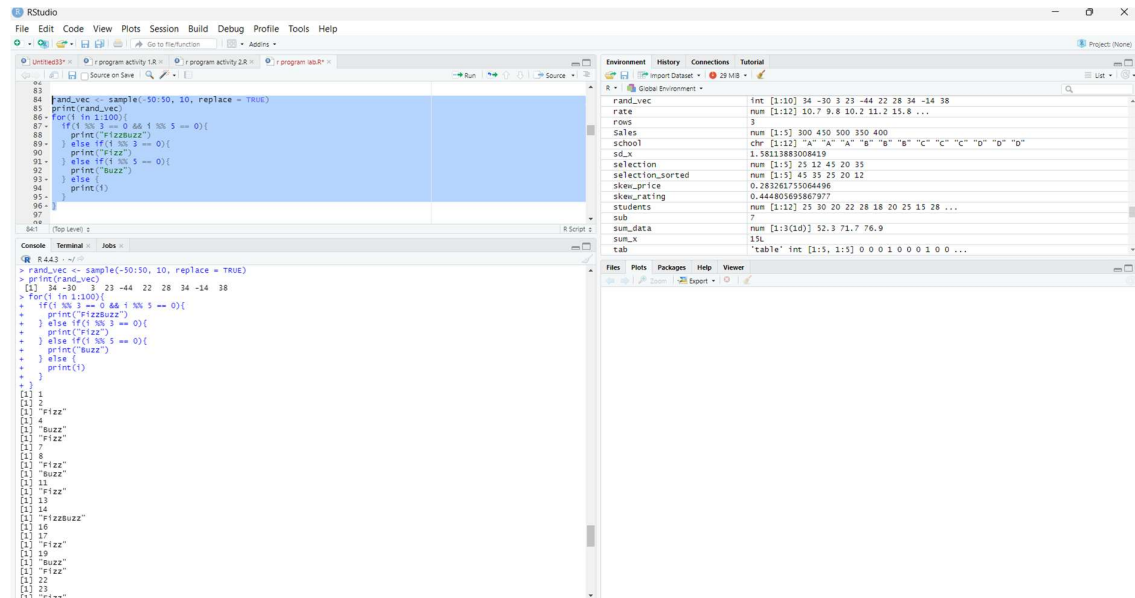
  } else {

    print(i)

  }

}
```

OUTPUT:

The screenshot shows the RStudio interface. The source editor on the left contains the R code for generating a random vector and performing a FizzBuzz loop. The console on the bottom left shows the execution output, including the random vector and the FizzBuzz results for numbers 1 through 23. The environment pane on the right lists the objects created during the session, such as 'rand_vec', 'rate', 'rows', 'Sales', 'school', 'sd_x', 'selection', 'selection_sorted', 'skew_price', 'skew_rating', 'students', 'sub', 'sum_data', 'sum_x', and 'tab'.

8. Generate random numbers from a normal distribution; count occurrences.

CODE:

```
set.seed(1)

x <- rnorm(20, mean = 0, sd = 1)
```

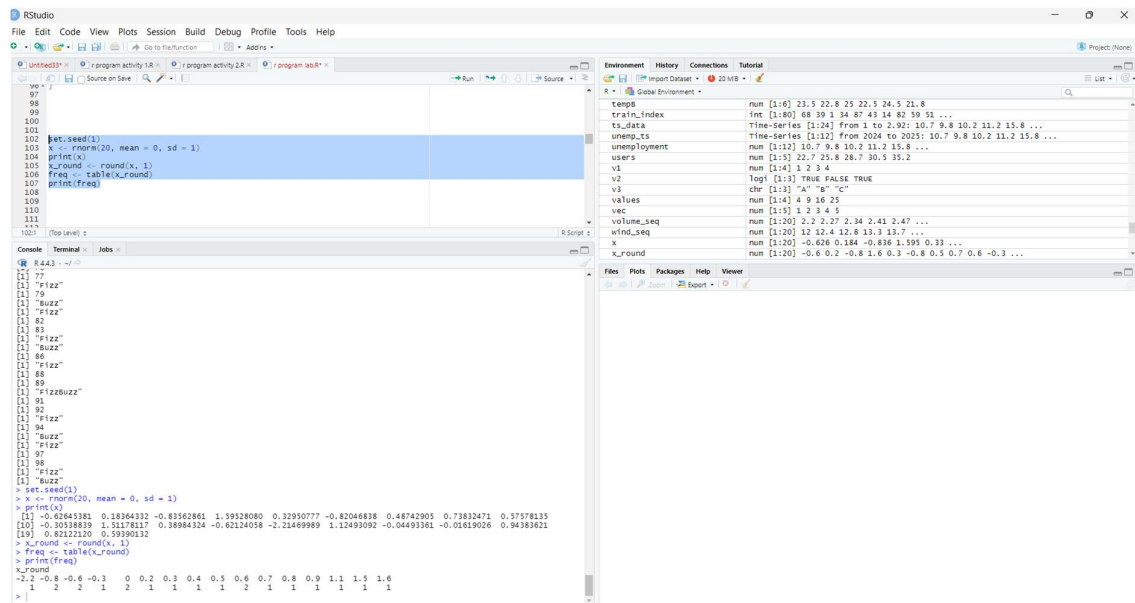
```
print(x)
```

```
x_round <- round(x, 1)
```

```
freq <- table(x_round)
```

```
print(freq)
```

OUTPUT:



9. Create empty plots with specified axis limits.

CODE:

```
plot(1, type = "n",
```

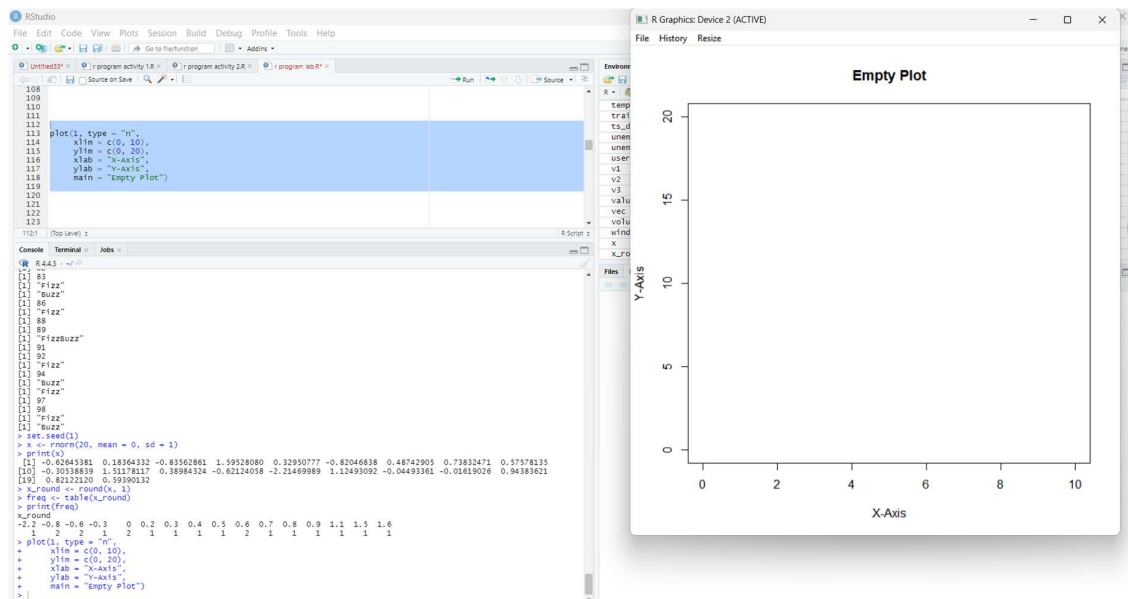
$$\mathbf{xlim} = \mathbf{c}(0, 10),$$
$$\text{ylim} = \text{c}(0, 20),$$

```
xlabel = "X-Axis",
```

```
ylab = "Y-Axis",
```

```
main = "Empty Plot")
```

OUTPUT:



10. Create and explore a data frame exam_data with name, score, attempts, and qualify fields. Perform extract, add row/column, sort, save to file.

CODE:

```

exam_data <- data.frame(
  name = c("niki", "navya", "Chandu", "Divya"),
  score = c(85, 72, 90, 65),
  attempts = c(1, 2, 1, 3),
  qualify = c(TRUE, TRUE, TRUE, FALSE)
)

print(exam_data)

print(exam_data$score)

print(exam_data[1:2, ])

new_row <- data.frame(
  name = "Esha",
  score = 88,
  attempts = 1,
  qualify = TRUE
)

exam_data <- rbind(exam_data, new_row)

exam_data$grade <- c("B", "C", "A", "D", "B")

print(exam_data)

```

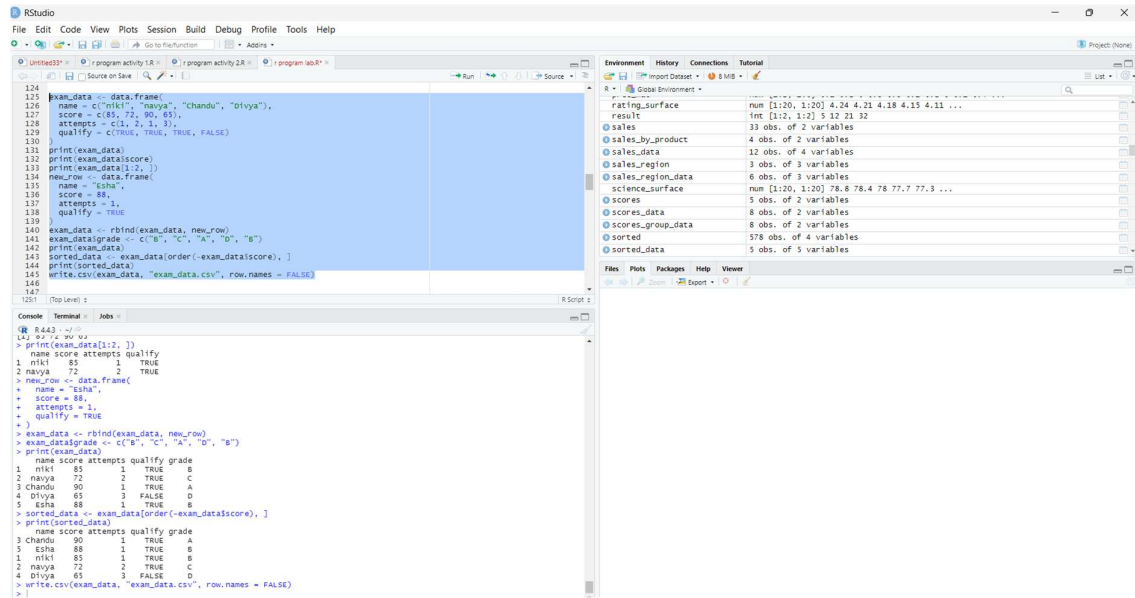


```
sorted_data <- exam_data[order(-exam_data$score), ]
```

```
print(sorted_data)
```

```
write.csv(exam_data, "exam_data.csv", row.names = FALSE)
```

OUTPUT:



The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for creating a data frame, adding a new row, sorting by score, and writing to a CSV file.
- Console:** Displays the execution output, showing the original data frame, the new row, the sorted data frame, and the successful CSV write operation.
- Environment:** Lists the objects in the global environment, including the sorted data frame.

```
# RStudio Source Editor Code
124
125 exam_data <- data.frame(
126   name = c("nikl", "navya", "Chandu", "Divya"),
127   score = c(85, 72, 90, 65),
128   attempts = c(1, 2, 1, 3),
129   qualify = c(TRUE, TRUE, TRUE, FALSE)
130 )
131 print(exam_data)
132 print(exam_data$score)
133 print(exam_data[2, ])
134 new_row <- data.frame(
135   name = "Esha",
136   score = 88,
137   attempts = 1,
138   qualify = TRUE
139 )
140 exam_data <- rbind(exam_data, new_row)
141 exam_data$grade <- c("B", "C", "A", "D", "B")
142 print(exam_data)
143 sorted_data <- exam_data[order(-exam_data$score), ]
144 print(sorted_data)
145 write.csv(exam_data, "exam_data.csv", row.names = FALSE)
146
147
```

Console Output:

```
R> # RStudio
R> [1] 0 0 0 0 0 0
R> print(exam_data[1:2, ])
  name score attempts qualify
1 nikl  85         1     TRUE
2 navya  72         2     TRUE
R> new_row <- data.frame(
+   name = "Esha",
+   score = 88,
+   attempts = 1,
+   qualify = TRUE
+ )
R> exam_data <- rbind(exam_data, new_row)
R> exam_data$grade <- c("B", "C", "A", "D", "B")
R> print(exam_data)
  name score attempts qualify grade
1 nikl  85         1     TRUE    B
2 navya  72         2     TRUE    C
3 Chandu 90         1     TRUE    A
4 Divya  65         3     FALSE    D
5 Esha  88         1     TRUE    B
R> sorted_data <- exam_data[order(-exam_data$score), ]
R> print(sorted_data)
  name score attempts qualify grade
3 Chandu 90         1     TRUE    A
5 Esha  88         1     TRUE    B
1 nikl  85         1     TRUE    B
2 navya  72         2     TRUE    C
4 Divya  65         3     FALSE    D
R> write.csv(exam_data, "exam_data.csv", row.names = FALSE)
R>
```

Environment Pane:

Object	Class	Attributes
rating_surface	num	[1:20, 1:20] 4.24 4.21 4.18 4.15 ...
result	int	[1:2, 1:2] 5 12 21 32
sales	33 obs. of 2 variables	
sales_by_product	4 obs. of 2 variables	
sales_data	12 obs. of 4 variables	
sales_region	3 obs. of 3 variables	
sales_region_data	6 obs. of 3 variables	
science_surface	num	[1:20, 1:20] 78.8 78.4 78 77.7 77.3 ...
scores	5 obs. of 2 variables	
scores_data	8 obs. of 2 variables	
scores_group_data	8 obs. of 2 variables	
sorted	578 obs. of 4 variables	
sorted_data	5 obs. of 5 variables	