

1

LAB-3

1. Perform the operation of combining matrices in R using cbind () and rbind() functions

The screenshot shows the RStudio IDE with the following content:

Source Panel:

```
RStudio
File Edit Code View Plot Session Build Debug Profile Tools Help
[Icons] [Add Plots] [Add Connections] [Add Connections] [Add Connections]
Source Terminal Background Jobs
# R - R4.0.2
> matrix1 <- matrix(1:5, nrow = 2, ncol = 5)
> matrix2 <- matrix(7:12, nrow = 2, ncol = 5)
>
> combined_by_column <- cbind(matrix1, matrix2)
> combined_by_row <- rbind(matrix1, matrix2)
>
> print("Combined by Column:")
[[1] "Combined by Column:"
> print(combined_by_column)
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    2    3    4    5    7    8    9
[2,]    6    7    8    9   10   11   12   13
>
> print("Combined by Row:")
[[1] "Combined by Row:"
> print(combined_by_row)
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    6    7    8
[3,]    7    8    9
[4,]   10   11   12
>
> }
```

Environment Panel:

Global Environment | 20 Rows | 12 Lines

data

combined_by_column	int [1:2, 1:6]	1 2 3 4 5 6 7 8 9 10
combined_by_row	int [1:4, 1:3]	1 2 3 4 5 6 7 8 9 10 11 12

matrix1
int [1:2, 1:5] 1 2 3 4 5 6

matrix2
int [1:2, 1:5] 7 8 9 10 11 12

2 .(i) Create vector of numeric, complex, logical and character with types of length 6.

(ii) Write a R program to add a new item g4 = "R Prog" to a given list.

The screenshot displays the RStudio interface. The console on the left shows the execution of R code that creates several vectors and a list. The Environment pane on the right shows the objects created in the global environment, including a list named 'my_list' containing the four vectors.

```

RStudio
File Edit View Plot Session Build Debug Profile Tools Help
[Icons] Add-on Packages
Console Terminal Background Jobs
> # R4.2
> numeric_vector <- c(1.1, 2.2, 3.3, 4.4, 5.5, 6.6)
> complex_vector <- c(1+2i, 3+4i, 5+6i, 7+8i, 9+10i, 11+12i)
> logical_vector <- c(TRUE, FALSE, TRUE, FALSE, TRUE, FALSE)
> character_vector <- c("apple", "banana", "cherry", "date", "elderberry", "fig")
>
> print("numeric_vector")
[1] "numeric_vector"
> print(numeric_vector)
[1] 1.1 2.2 3.3 4.4 5.5 6.6
>
> print("complex_vector")
[1] "complex_vector"
> print(complex_vector)
[1] 1+2i 3+4i 5+6i 7+8i 9+10i 11+12i
>
> print("logical_vector")
[1] "logical_vector"
> print(logical_vector)
[1] TRUE FALSE TRUE FALSE TRUE FALSE
>
> print("character_vector")
[1] "character_vector"
> print(character_vector)
[1] "apple" "banana" "cherry" "date" "elderberry" "fig"
>
> my_list <- list(a = 1, b = 2, c = 3)
> my_list[[2]] <- "R mag"
> print("updated list")
[1] "updated list"
> print(my_list)
[[
  [1] 1
]
[[
  [1] 2
]
[[
  [1] 3
]
]
>
> print("R mag")
[1] "R mag"

```

Environment History Connections Tutorial

Object	Type	Value
my_list	List of 3	
numeric_vector	dbl [1:6]	1.1 2.2 3.3 4.4 5.5 6.6
complex_vector	cplx [1:6]	1+2i 3+4i 5+6i ...
logical_vector	lgl [1:6]	TRUE FALSE TRUE FALSE TRUE FALSE
character_vector	chr [1:6]	1.1 2.2 3.3 4.4 5.5 6.6

File Edit Plot Session Build Debug Profile Tools Help

3. The price of one kg of rice is Rs. 40.75 and one kg of sugar is Rs. 30. Write R program to get the total amount of 2kg rice and 5kg sugar purchase

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

```
R - R442 - v1.0
> rice_price_per_kg <- 40.75
> sugar_price_per_kg <- 30
> rice_quantity <- 2
> sugar_quantity <- 5
> total_amount <- (rice_price_per_kg * rice_quantity) + (sugar_price_per_kg * sugar_quantity)
> print(paste("Total amount for 2kg rice and 5kg sugar: Rs. ", total_amount))
[1] "Total amount for 2kg rice and 5kg sugar: Rs. 233.5"
>
```

The console on the right shows the output of the script:

```
Environment: Global Environment
Date: 2023-08-08 10:10:10
Data:
  rice_price_per_kg 40.75
  rice_quantity    2
  sugar_price_per_kg 30
  sugar_quantity    5
  total_amount     233.5
```

+

4. Write a R code to create a 3x4 matrix with 12 random numbers between 1-100; have the matrix be filled our row-by-row, instead of column-by-column. Name the columns of the matrix uno, dos, tres, cuatro, and the rows x, y, z. Scale the matrix by 10 and save the result

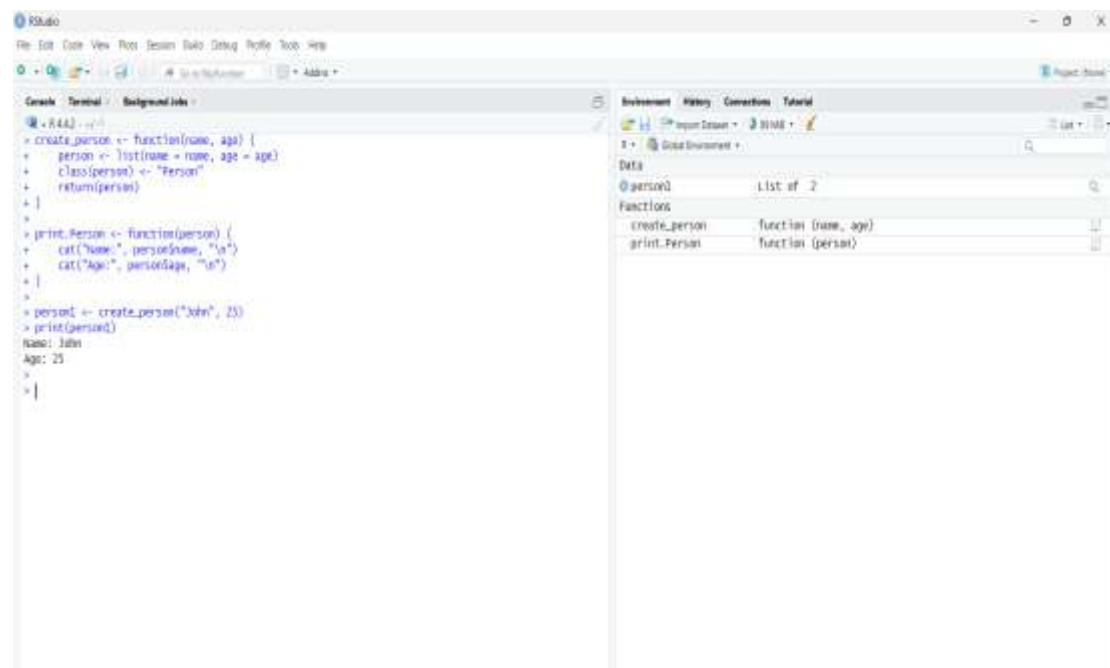
The screenshot shows the RStudio interface. The script editor on the left contains the following R code:

```
R - R442 - v1.0
> set.seed(123)
> matrix_data <- matrix(sample(1:100, 12, replace = TRUE), nrow = 3, ncol = 4, byrow = TRUE)
> colnames(matrix_data) <- c("uno", "dos", "tres", "cuatro")
> rownames(matrix_data) <- c("x", "y", "z")
> scaled_matrix <- matrix_data * 10
> print("Original Matrix:")
[1] "Original Matrix:"
> print(matrix_data)
     dos tres cuatro
x  31  78  53   14
y  47  42  50   43
z  14  25  90   91
[1] "Scaled Matrix by 10:"
[1] "Scaled Matrix by 10:"
> print(scaled_matrix)
     dos tres cuatro
x 310 780 530 140
y 470 420 500 430
z 140 250 900 910
>
```

The console on the right shows the output of the script:

```
Environment: Global Environment
Date: 2023-08-08 10:10:10
Data:
  matrix_data      dos tres cuatro
scaled_matrix     dos tres cuatro
x 310 780 530 140
y 470 420 500 430
z 140 250 900 910
```

5. Demonstrate the creation of S3 class in R.



The screenshot shows the RStudio interface with the following content:

Console:

```
> create_person <- function(name, age) {  
+   person <- list(name = name, age = age)  
+   class(person) <- "Person"  
+   return(person)  
+ }  
+  
+ print.Person <- function(person) {  
+   cat("Name: ", person$name, "\n")  
+   cat("Age: ", person$age, "\n")  
+ }  
+  
+ person <- create_person("John", 25)  
+ print(person)  
Name: John  
Age: 25  
+  
+ |
```

Environment:

Object	Class
person	List of 2
create_person	function (name, age)
print.Person	function (person)