

## Lab Experiments (Day 3)

1. Write a R program to create an array of two 3x3 matrices each with 3 rows and 3 columns from two given two vectors. Print the second row of the second matrix of the array and the element in the 3rd row and 3rd column of the 1st matrix

```
> print("Two vectors of different lengths:")
[1] "Two vectors of different lengths:"
> v1 = c(1,3,4,5)
> v2 = c(10,11,12,13,14,15)
> print(v1)
[1] 1 3 4 5
> print(v2)
[1] 10 11 12 13 14 15
> result = array(c(v1,v2),dim = c(3,3,2))
> print("New array:")
[1] "New array:"
> print(result)
, , 1
      [,1] [,2] [,3]
[1,]    1    5   12
[2,]    3   10   13
[3,]    4   11   14

, , 2
      [,1] [,2] [,3]
[1,]   15    4   11
[2,]    1    5   12
[3,]    3   10   13

> print("The second row of the second matrix of the array:")
[1] "The second row of the second matrix of the array:"
> print(result[2,,2])
[1] 1 5 12
> print("The element in the 3rd row and 3rd column of the 1st matrix:")
[1] "The element in the 3rd row and 3rd column of the 1st matrix:"
> print(result[3,3,1])
[1] 14
> |
```

2. Write a R program to combine three arrays so that the first row of the first array is followed by the first row of the second array and then first row of the third array

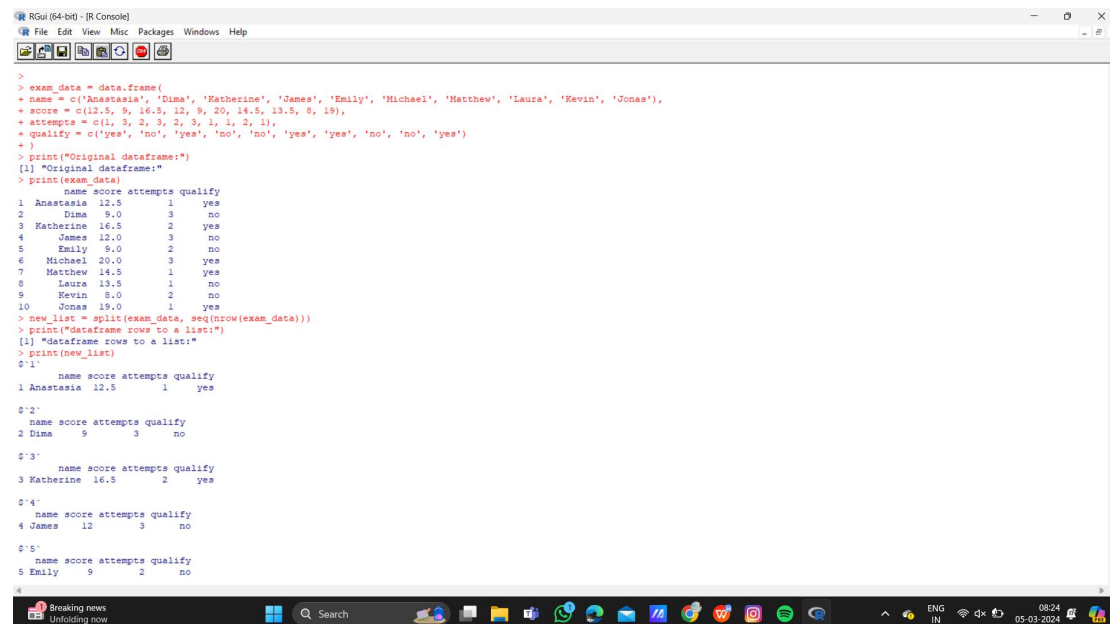
```
> num1 = rbind(rep("A",3), rep("B",3), rep("C",3))
> print("num1")
[1] "num1"
> print(num1)
      [,1] [,2] [,3]
[1,] "A"  "A"  "A"
[2,] "B"  "B"  "B"
[3,] "C"  "C"  "C"
> num2 = rbind(rep("P",3), rep("Q",3), rep("R",3))
> print("num2")
[1] "num2"
> print(num2)
      [,1] [,2] [,3]
[1,] "P"  "P"  "P"
[2,] "Q"  "Q"  "Q"
[3,] "R"  "R"  "R"
> num3 = rbind(rep("X",3), rep("Y",3), rep("Z",3))
> print("num3")
[1] "num3"
> print(num3)
      [,1] [,2] [,3]
[1,] "X"  "X"  "X"
[2,] "Y"  "Y"  "Y"
[3,] "Z"  "Z"  "Z"
> a = matrix(t(cbind(num1,num2,num3)),ncol=3, byrow=T)
> print("Combine three arrays, taking one row from each one by one:")
[1] "Combine three arrays, taking one row from each one by one:"
> print(a)
      [,1] [,2] [,3]
[1,] "A"  "A"  "A"
[2,] "P"  "P"  "P"
[3,] "X"  "X"  "X"
[4,] "B"  "B"  "B"
[5,] "Q"  "Q"  "Q"
[6,] "Y"  "Y"  "Y"
[7,] "C"  "C"  "C"
[8,] "R"  "R"  "R"
[9,] "Z"  "Z"  "Z"
> |
```

3. Write a R program to create an array using four given columns, three given rows, and two given tables and display the content of the array  
Write a R program to create a two-dimensional 5x3 array of sequence of even integers greater than 50

```
> array1 = array(1:30, dim=c(3,5,2))
> print(array1)
, , 1
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    4    7   10   13
[2,]    2    5    8   11   14
[3,]    3    6    9   12   15

, , 2
     [,1] [,2] [,3] [,4] [,5]
[1,]   16   19   22   25   28
[2,]   17   20   23   26   29
[3,]   18   21   24   27   30
```

4. Create below data frame exam\_data = data. Frame( name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'), score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19), attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1), qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes') ) a. Write a R program to extract 3rd and 5th rows with 1st and 3rd columns from a given data frame b. Write a R program to add a new column named country in a given data frame Country<-c("USA","USA","USA","USA","UK","USA","USA","India","USA","USA")



```
> exam_data = data.frame(
+ name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
+ score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
+ attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
+ qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
+ )
> print("Original dataframe:")
[1] "Original dataframe:"
> print(exam_data)
  name score attempts qualify
1 Anastasia 12.5      1    yes
2 Dima      9.0      3     no
3 Katherine 16.5      2    yes
4 James    12.0      3     no
5 Emily     9.0      2     no
6 Michael  20.0      3    yes
7 Matthew  14.5      1    yes
8 Laura    13.5      1     no
9 Kevin     8.0      2     no
10 Jonas   19.0      1    yes
> new_list = split(exam_data, seq(nrow(exam_data)))
> print("dataframe rows to a list:")
[1] "dataframe rows to a list:"
> print(new_list)
$1:
  name score attempts qualify
1 Anastasia 12.5      1    yes

$2:
  name score attempts qualify
2 Dima      9.0      3     no

$3:
  name score attempts qualify
3 Katherine 16.5      2    yes

$4:
  name score attempts qualify
4 James    12.0      3     no

$5:
  name score attempts qualify
5 Emily     9.0      2     no
```

5. Write a R program to add new row(s) to an existing data frame `new_exam_data = data.frame(name = c('Robert', 'Sophia'), score = c(10.5, 9), attempts = c(1, 3), qualify = c('yes', 'no'))` d. Write a R program to sort a given data frame by name and score e. Write a R program to save the information of a data frame in a file and display the information of the file.

```

RGui (64-bit) - [R Console]
File Edit View Misc Packages Windows Help

> exam_data = data.frame(
+ name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'),
+ score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
+ attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),
+ qualify = c('yes', 'no', 'yes', 'no', 'yes', 'yes', 'no', 'no', 'yes', 'yes')
+ )
> print("Original dataframe:")
[1] "Original dataframe:"
> print(exam_data)
  name score attempts qualify
1 Anastasia 12.5      1    yes
2 Dima      9.0      3     no
3 Katherine 16.5      2    yes
4 James    12.0      3     no
5 Emily     9.0      2     no
6 Michael  20.0      3    yes
7 Matthew  14.5      1    yes
8 Laura    13.5      1     no
9 Kevin     8.0      2     no
10 Jonas   19.0      1    yes
> new_exam_data = data.frame(
+ name = c('Robert', 'Sophia'),
+ score = c(10.5, 9),
+ attempts = c(1, 3),
+ qualify = c('yes', 'no')
+ )
> exam_data = rbind(exam_data, new_exam_data)
> print("After adding new row(s) to an existing data frame:")
[1] "After adding new row(s) to an existing data frame:"
> print(exam_data)
  name score attempts qualify
1 Anastasia 12.5      1    yes
2 Dima      9.0      3     no
3 Katherine 16.5      2    yes
4 James    12.0      3     no
5 Emily     9.0      2     no
6 Michael  20.0      3    yes
7 Matthew  14.5      1    yes
8 Laura    13.5      1     no
9 Kevin     8.0      2     no
10 Jonas   19.0      1    yes
11 Robert  10.5      1    yes
12 Sophia   9.0      3     no
>

```

6. Write a R program to call the (built-in) dataset `airquality`. Check whether it is a data frame or not? Order the entire data frame by the first and second column. remove the variables 'Solar.R' and 'Wind' and display the data frame

```

RGui (64-bit) - [R Console]
File Edit View Misc Packages Windows Help

> data = airquality
> print("Original data: Daily air quality measurements in New York, May to September 1973.")
[1] "Original data: Daily air quality measurements in New York, May to September 1973."
> print(class(data))
[1] "data.frame"
> print(head(data, 10))
  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   66     5    5
6    28     NA 14.9   66     5    6
7    23    296  8.6   65     5    7
8    19     99 13.0   59     5    8
9     8     19 20.1   61     5    9
10   NA    194  8.6   69     5   10
> result = data[order(data[,1]),]
> print("Order the entire data frame by the first and second column:")
[1] "Order the entire data frame by the first and second column:"
> print(result)
  Ozone Solar.R Wind Temp Month Day
21    1     8  9.7   59     5   21
23    4    25  9.7   61     5   23
18    6    78 18.4   57     5   18
11    7    NA  6.9   74     5   11
76    7    48 14.3   80     7   15
147   7    49 10.3   69     9   24
9     8    19 20.1   61     5    9
94    9    24 13.8   81     8    2
114   9    36 14.3   72     8   22
137   9    24 10.9   71     9   14
73   10   264 14.3   73     7   12
13   11   280  9.2   66     5   13
20   11    44  9.7   62     5   20
22   11   320 16.6   73     5   22
3    12   149 12.6   74     5    3
50   12   120 11.5   73     6   19
51   13   137 10.3   76     6   20
138   13   112 11.5   71     9   15
141   13    27 10.3   76     9   18
144   13   238 12.6   64     9   21
14    14   274 10.9   68     5   14
16    14   334 11.5   64     5   16

```

7 Write a R program to create a factor corresponding to height of women data set, which inbuilt in R, contains height and weights for a sample of women.

```
> data = women
> print("Women data set of height and weights:")
[1] "Women data set of height and weights:"
> print(data)
  height weight
1     58   115
2     59   117
3     60   120
4     61   123
5     62   126
6     63   129
7     64   132
8     65   135
9     66   139
10    67   142
11    68   146
12    69   150
13    70   154
14    71   159
15    72   164
> height_f = cut(women$height,3)
> print("Factor corresponding to height:")
[1] "Factor corresponding to height:"
> print(table(height_f))
height_f
[58,62.7] [62.7,67.3] [67.3,72]
      5           5           5
```

8. Write a R program to extract the five of the levels of factor created from a random sample from the LETTERS (Part of the base R distribution.)

```
> L = sample(LETTERS,size=50,replace=TRUE)
> print("Original data:")
[1] "Original data:"
> print(L)
[1] "p" "e" "a" "g" "e" "p" "h" "a" "x" "i" "t" "e" "p" "i" "u" "e" "z" "u" "r" "e" "a" "c" "o" "i" "y" "h" "e" "s" "e" "k" "e" "s" "u" "l" "h" "j" "g" "d" "e" "u" "s" "g" "r" "e" "k" "e" "u" "h" "j" "u" "l" "e" "u" "h" "l" "e" "p" "h" "o" "a" "h" "i" "t" "e" "p" "e" "o" "e" "i"
[47] "h" "e" "u" "e" "l" "e"
> F = factor(L)
> print("Original factors:")
[1] "Original factors:"
> print(F)
[1] G I H F M X T P I U Z V R C O Y N S K S V L J D O V R K E V J L U N L B H O O A T F E O B N Z L K
Levels: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
> print("Only five of the levels")
[1] "Only five of the levels"
> print(table(L[1:5]))

G H I P
1 1 2 1
```

9. Iris dataset is a very famous dataset in almost all data mining, machine learning courses, and it has been an R build-in dataset. The dataset consists of 50 samples from each of three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor). Four features(variables) were measured from each sample, they are the length and the width of sepal and petal, in centimetres. Perform the following EDA steps.

- (i) Find dimension, Structure, Summary statistics, Standard Deviation of all features.
- (ii) Find mean and standard deviation of features grouped by three species of Iris flowers (Iris setosa, Iris virginica and Iris versicolor)
- (iii) Find quantile value of sepal width and length
- (iv) Create new data frame named iris1 which has a new column name Sepal.Length.Cate that categorizes "Sepal.Length" by quantile
- (v) Average value of numerical variables by two categorical variables: Species and Sepal.Length.Cate
- (vi) Average mean value of numerical variables by Species and Sepal.Length.Cate
- (vii) Create Pivot Table based on Species and Sepal.Length.Cate.

```
RGui (64-bit) - R Console
File Edit View Misc Packages Windows Help

>
> # Load Iris dataset
> data(iris)
>
> # (i) Find dimension, Structure, Summary statistics, Standard Deviation of all Features
> dim(iris) # Dimension
[1] 150 5
> str(iris) # Structure
'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 ...
> summary(iris) # Summary statistics
      Sepal.Length      Sepal.Width      Petal.Length      Petal.Width      Species
Min.   4.300000    Min.   2.000000    Min.   1.000000    Min.   0.100000    setosa      :50
1st Qu.:5.1000    1st Qu.:2.8000    1st Qu.:1.6000    1st Qu.:0.1000    versicolor:50
Median :5.8000    Median :3.0000    Median :4.3500    Median :1.3000    virginica  :50
Mean   :5.8433    Mean   :3.0577    Mean   :4.7500    Mean   :1.1999
3rd Qu.:6.4000    3rd Qu.:3.3000    3rd Qu.:5.1000    3rd Qu.:1.8000
Max.   :7.9000    Max.   :4.4000    Max.   :6.9000    Max.   :2.5000
> sd(iris[, 1:4]) # Standard Deviation of all features
Sepal.Length Sepal.Width Petal.Length Petal.Width
0.8280661    0.4358663    1.7652982    0.7622377
>
> # (ii) Find mean and standard deviation of features grouped by three species of Iris flowers
> aggregate(~ Species, data = iris, FUN = function(x) c(mean = mean(x), sd = sd(x)))
      Species Sepal.Length.mean Sepal.Length.sd Sepal.Width.mean Sepal.Width.sd Petal.Length.mean Petal.Length.sd Petal.Width.mean Petal.Width.sd
1 setosa      5.0060000         0.3524897         3.4280000         0.3790644         4.4620000         0.1736640         0.2460000         0.1053856
2 versicolor  5.9360000         0.5161711         2.7700000         0.3137983         4.2600000         0.4699110         1.3260000         0.1977527
3 virginica   6.5880000         0.6358796         2.9740000         0.3224966         5.5520000         0.5518947         2.0260000         0.2746501
>
> # (iii) Find quantile value of sepal width and length
> quantile(iris$Sepal.Width)
 0% 25% 50% 75% 100%
2.0 2.8 3.0 3.3 4.4
> quantile(iris$Sepal.Length)
 0% 25% 50% 75% 100%
4.3 5.1 5.8 6.4 7.9
>
> # (iv) Create new data frame named iris1 which has a new column named Sepal.Length.Cate that categorizes "Sepal.Length" by quantile
> iris1 <- iris
> iris1$Sepal.Length.Cate <- cut(iris1$Sepal.Length, breaks = quantile(iris1$Sepal.Length), labels = FALSE)
>
Species Sepal.Length.mean Sepal.Length.sd Sepal.Width.mean Sepal.Width.sd Petal.Length.mean Petal.Length.sd Petal.Width.mean Petal.Width.sd
1 setosa      5.0060000         0.3524897         3.4280000         0.3790644         4.4620000         0.1736640         0.2460000         0.1053856
2 versicolor  5.9360000         0.5161711         2.7700000         0.3137983         4.2600000         0.4699110         1.3260000         0.1977527
3 virginica   6.5880000         0.6358796         2.9740000         0.3224966         5.5520000         0.5518947         2.0260000         0.2746501
>
> # (iii) Find quantile value of sepal width and length
> quantile(iris1$Sepal.Width)
 0% 25% 50% 75% 100%
2.0 2.8 3.0 3.3 4.4
> quantile(iris1$Sepal.Length)
 0% 25% 50% 75% 100%
4.3 5.1 5.8 6.4 7.9
>
> # (iv) Create new data frame named iris1 which has a new column named Sepal.Length.Cate that categorizes "Sepal.Length" by quantile
> iris1 <- iris
> iris1$Sepal.Length.Cate <- cut(iris1$Sepal.Length, breaks = quantile(iris1$Sepal.Length), labels = FALSE)
>
> # (v) Average value of numerical variables by two categorical variables: Species and Sepal.Length.Cate
> aggregate(~ Species + Sepal.Length.Cate, data = iris1, FUN = mean)
      Species Sepal.Length.Cate Sepal.Length Sepal.Width Petal.Length Petal.Width
1 setosa      1 4.854286      3.300000      1.465714      0.2457143
2 versicolor  1 5.000000      2.300000      3.275000      1.0250000
3 virginica   1 4.900000      2.500000      4.500000      1.7000000
4 setosa      2 5.435714      3.778571      1.478571      0.2571429
5 versicolor  2 5.600000      2.705000      4.055000      1.2400000
6 virginica   2 5.740000      2.700000      5.040000      2.0400000
7 versicolor  3 6.135294      2.832994      4.511765      1.4294118
8 virginica   3 6.238889      2.900000      5.283333      1.9222222
9 versicolor  4 6.722222      3.000000      4.677778      1.4555556
10 virginica  4 7.057692      3.096154      5.876923      2.1076923
>
> # (vi) Average mean value of numerical variables by Species and Sepal.Length.Cate
> aggregate(~ Species + Sepal.Length.Cate, data = iris1, FUN = function(x) mean(mean(x)))
      Species Sepal.Length.Cate Sepal.Length Sepal.Width Petal.Length Petal.Width
1 setosa      1 4.854286      3.300000      1.465714      0.2457143
2 versicolor  1 5.000000      2.300000      3.275000      1.0250000
3 virginica   1 4.900000      2.500000      4.500000      1.7000000
4 setosa      2 5.435714      3.778571      1.478571      0.2571429
5 versicolor  2 5.600000      2.705000      4.055000      1.2400000
6 virginica   2 5.740000      2.700000      5.040000      2.0400000
7 versicolor  3 6.135294      2.832994      4.511765      1.4294118
8 virginica   3 6.238889      2.900000      5.283333      1.9222222
9 versicolor  4 6.722222      3.000000      4.677778      1.4555556
10 virginica  4 7.057692      3.096154      5.876923      2.1076923
```