**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**COURSE CODE / SUBJECT: ITA 0448 / STATISTICS WITH R PROGRAMMING FOR VECTORIZED EXPRESSIONS**

**DAY 2 – LAB ASSESSMENT**

**Reg No:192124049**

**Name:SK KHAJAMAINUDDIN**

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.

> #https://bit.ly/2QkvW10> 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.

> 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.** Write a R program to create a two-dimensional 5x3 array of sequence of even integers greater than 50.

> a <- array(seq(from = 50, length.out = 15, by = 2), c(5, 3))> print("Content of the array:")[1] "Content of the array:"> print("5×3 array of sequence of even integers greater than 50:")[1] "5×3 array of sequence of even integers greater than 50:"> print(a) [,1] [,2] [,3]

[1,] 50 60 70

[2,] 52 62 72

[3,] 54 64 74

[4,] 56 66 76

[5,] 58 68 78

**Use Below Data frame from question 5 to 9**

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')

)

5. Write a R program to extract 3rd and 5th rows with 1st and 3rd columns from a given 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')+ )> 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> print("Extract 3rd and 5th rows with 1st and 3rd columns :")[1] "Extract 3rd and 5th rows with 1st and 3rd columns :"> result = exam\_data[c(3,5),c(1,3)]> print(result) name attempts

3 Katherine 2

5 Emily 2

6. 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> print("New data frame after adding the 'country' column:")[1] "New data frame after adding the 'country' column:"> exam\_data$country = c("USA","USA","USA","USA","USA","USA","USA","USA","USA","USA")> print(exam\_data) name score attempts qualify country

1 Anastasia 12.5 1 yes USA

2 Dima 9.0 3 no USA

3 Katherine 16.5 2 yes USA

4 James 12.0 3 no USA

5 Emily 9.0 2 no USA

6 Michael 20.0 3 yes USA

7 Matthew 14.5 1 yes USA

8 Laura 13.5 1 no USA

9 Kevin 8.0 2 no USA

10 Jonas 19.0 1 yes USA

7. Write a R program to add new row(s) to an existing data frame

new\_exam\_data = data.frame(name = c(&#39;Robert&#39;, &#39;Sophia&#39;),score = c(10.5, 9), attempts = c(1,

3),qualify = c(&#39;yes&#39;, &#39;no&#39;))  
  
  
  
> 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\_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

8. Write a R program to sort a given data frame by name and score

> 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> print("dataframe after sorting 'name' and 'score' columns:")[1] "dataframe after sorting 'name' and 'score' columns:"> exam\_data = exam\_data[with(exam\_data, order(name, score)), ]> print(exam\_data) name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 3 no

5 Emily 9.0 2 no

4 James 12.0 3 no

10 Jonas 19.0 1 yes

3 Katherine 16.5 2 yes

9 Kevin 8.0 2 no

8 Laura 13.5 1 no

7 Matthew 14.5 1 yes

6 Michael 20.0 3 yes

9.  Write a R program to save the information of a data frame in a file and display the information of the file.

> 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> save(exam\_data,file="data.rda")> load("data.rda")> file.info("data.rda") size isdir mode mtime

data.rda 302 FALSE 666 2023-04-30 10:47:54

ctime atime exe

data.rda 2023-04-30 10:47:54 2023-04-30 10:47:54 no

10. 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.

> 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 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> 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 11 290 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

148 14 20 16.6 63 9 25

151 14 191 14.3 75 9 28

12 16 256 9.7 69 5 12

82 16 7 6.9 74 7 21

95 16 77 7.4 82 8 3

143 16 201 8.0 82 9 20

4 18 313 11.5 62 5 4

15 18 65 13.2 58 5 15

140 18 224 13.8 67 9 17

152 18 131 8.0 76 9 29

8 19 99 13.8 59 5 8

49 20 37 9.2 65 6 18

87 20 81 8.6 82 7 26

130 20 252 10.9 80 9 7

153 20 223 11.5 68 9 30

47 21 191 14.9 77 6 16

113 21 259 15.5 77 8 21

132 21 230 10.9 75 9 9

135 21 259 15.5 76 9 12

108 22 71 10.3 77 8 16

7 23 299 8.6 65 5 7

28 23 13 12.0 67 5 28

44 23 148 8.0 82 6 13

110 23 115 7.4 76 8 18

131 23 220 10.3 78 9 8

145 23 14 9.2 71 9 22

133 24 259 9.7 73 9 10

142 24 238 10.3 68 9 19

74 27 175 14.9 81 7 13

6 28 NA 14.9 66 5 6

105 28 273 11.5 82 8 13

136 28 238 6.3 77 9 13

38 29 127 9.7 82 6 7

19 30 322 11.5 68 5 19

149 30 193 6.9 70 9 26

111 31 244 10.9 78 8 19

24 32 92 12.0 61 5 24

64 32 236 9.2 81 7 3

129 32 92 15.5 84 9 6

17 34 307 12.0 66 5 17

78 35 274 10.3 82 7 17

97 35 NA 7.4 85 8 5

2 36 118 8.0 72 5 2

146 36 139 10.3 81 9 23

31 37 279 7.4 76 5 31

48 37 284 20.7 72 6 17

41 39 323 11.5 87 6 10

93 39 83 6.9 81 8 1

67 40 314 10.9 83 7 6

1 41 190 7.4 67 5 1

104 44 192 11.5 86 8 12

112 44 190 10.3 78 8 20

134 44 236 14.9 81 9 11

29 45 252 14.9 81 5 29

116 45 212 9.7 79 8 24

139 46 237 6.9 78 9 16

128 47 95 7.4 87 9 5

77 48 260 6.9 81 7 16

63 49 248 9.2 85 7 2

90 50 275 7.4 86 7 29

88 52 82 12.0 86 7 27

92 59 254 9.2 81 7 31

109 59 51 6.3 79 8 17

79 61 285 6.3 84 7 18

81 63 220 11.5 85 7 20

66 64 175 4.6 83 7 5

91 64 253 7.4 83 7 30

106 65 157 9.7 80 8 14

98 66 NA 4.6 87 8 6

40 71 291 13.8 90 6 9

118 73 215 8.0 86 8 26

126 73 183 2.8 93 9 3

120 76 203 9.7 97 8 28

68 77 276 5.1 88 7 7

96 78 NA 6.9 86 8 4

125 78 197 5.1 92 9 2

80 79 187 5.1 87 7 19

85 80 294 8.6 86 7 24

89 82 213 7.4 88 7 28

122 84 237 6.3 96 8 30

71 85 175 7.4 89 7 10

123 85 188 6.3 94 8 31

100 89 229 10.3 90 8 8

127 91 189 4.6 93 9 4

124 96 167 6.9 91 9 1

69 97 267 6.3 92 7 8

70 97 272 5.7 92 7 9

86 108 223 8.0 85 7 25

101 110 207 8.0 90 8 9

30 115 223 5.7 79 5 30

121 118 225 2.3 94 8 29

99 122 255 4.0 89 8 7

62 135 269 4.1 84 7 1

117 168 238 3.4 81 8 25

5 NA NA 14.3 56 5 5

10 NA 194 8.6 69 5 10

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

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

11. Write a R program to create a factor corresponding to height of women data set , which inbuild 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

12. 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] "R" "B" "K" "X" "Y" "H" "Y" "J" "E" "D" "W" "V" "E" "Q"

[15] "F" "I" "W" "Q" "A" "E" "M" "D" "R" "E" "N" "R" "J" "J"

[29] "P" "N" "C" "B" "V" "H" "H" "J" "Y" "K" "C" "M" "Q" "Q"

[43] "Q" "Y" "P" "M" "C" "G" "F" "Z"> f = factor(L)> print("Original factors:")[1] "Original factors:"> print(f) [1] R B K X Y H Y J E D W V E Q F I W Q A E M D R E N R J J P

[30] N C B V H H J Y K C M Q Q Q Y P M C G F Z

Levels: A B C D E F G H I J K M N P Q R V W X Y Z> print("Only five of the levels")[1] "Only five of the levels"> print(table(L[1:5]))

B K R X Y

1 1 1 1 1

13. **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 groped 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 have  a new column name **Sepal.Length.Cate** that categorizes “Sepal.Length” by quantile

(V) Average value of numerical varialbes by two categorical variables: Species and Sepal.Length.Cate:

(vi) Average mean value of numerical varialbes by Species and Sepal.Length.Cate

(vii)Create Pivot Table based on Species and Sepal.Length.Cate.

> data(iris)> dim(iris) [1] 150 5> 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 ...> summary(iris) Sepal.Length Sepal.Width Petal.Length

Min. :4.300 Min. :2.000 Min. :1.000

1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600

Median :5.800 Median :3.000 Median :4.350

Mean :5.843 Mean :3.057 Mean :3.758

3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100

Max. :7.900 Max. :4.400 Max. :6.900

Petal.Width Species

Min. :0.100 setosa :50

1st Qu.:0.300 versicolor:50

Median :1.300 virginica :50

Mean :1.199

3rd Qu.:1.800

Max. :2.500 > sd(iris$Sepal.Length)[1] 0.8280661> sd(iris$Sepal.Width) [1] 0.4358663> sd(iris$Petal.Length) [1] 1.765298> sd(iris$Petal.Width) [1] 0.7622377> > > aggregate(. ~ Species, data = iris, mean) # mean of all features grouped by Species Species Sepal.Length Sepal.Width Petal.Length Petal.Width

1 setosa 5.006 3.428 1.462 0.246

2 versicolor 5.936 2.770 4.260 1.326

3 virginica 6.588 2.974 5.552 2.026> aggregate(. ~ Species, data = iris, sd) # standard deviation of all features grouped by Species Species Sepal.Length Sepal.Width Petal.Length Petal.Width

1 setosa 0.3524897 0.3790644 0.1736640 0.1053856

2 versicolor 0.5161711 0.3137983 0.4699110 0.1977527

3 virginica 0.6358796 0.3224966 0.5518947 0.2746501> > quantile(iris$Sepal.Width) # quantile of Sepal.Width 0% 25% 50% 75% 100%

2.0 2.8 3.0 3.3 4.4 > quantile(iris$Sepal.Length) # quantile of Sepal.Length 0% 25% 50% 75% 100%

4.3 5.1 5.8 6.4 7.9 > > > iris1 <- iris> iris1$Sepal.Length.Cate <- cut(iris$Sepal.Length, breaks = quantile(iris$Sepal.Length))> head(iris1) 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

Sepal.Length.Cate

1 (4.3,5.1]

2 (4.3,5.1]

3 (4.3,5.1]

4 (4.3,5.1]

5 (4.3,5.1]

6 (5.1,5.8]> > > aggregate(. ~ Species + Sepal.Length.Cate, data = iris1, mean) Species Sepal.Length.Cate Sepal.Length Sepal.Width

1 setosa (4.3,5.1] 4.854286 3.300000

2 versicolor (4.3,5.1] 5.000000 2.300000

3 virginica (4.3,5.1] 4.900000 2.500000

4 setosa (5.1,5.8] 5.435714 3.778571

5 versicolor (5.1,5.8] 5.600000 2.705000

6 virginica (5.1,5.8] 5.740000 2.700000

7 versicolor (5.8,6.4] 6.135294 2.835294

8 virginica (5.8,6.4] 6.238889 2.900000

9 versicolor (6.4,7.9] 6.722222 3.000000

10 virginica (6.4,7.9] 7.057692 3.096154

Petal.Length Petal.Width

1 1.465714 0.2457143

2 3.275000 1.0250000

3 4.500000 1.7000000

4 1.478571 0.2571429

5 4.055000 1.2400000

6 5.040000 2.0400000

7 4.511765 1.4294118

8 5.283333 1.9222222

9 4.677778 1.4555556

10 5.876923 2.1076923> library(dplyr)

1. Titanic Casualties – Use the standard ‘Titanic’ dataset which is part of R Base to answer  
   the following questions.  
   (i). Use an appropriate apply function to get the sum of males vs females aboard.

> gender\_count <- apply(Titanic, c(3,4), sum)> > # Print the gender count> gender\_count Survived

Age No Yes

Child 52 57

Adult 1438 654

(ii). Get a table with the sum of survivors vs sex.

# Get the sum of survivors vs sex using table function

survivors\_by\_sex <- table(Titanic$Survived, Titanic$Sex)

# Print the survivors by sex

survivors\_by\_sex

Female Male

No 81 468

Yes 233 109

(iii). Get a table with the sum of passengers by sex vs age

# Get the sum of passengers by sex vs age using table function

passengers\_by\_sex\_age <- table(Titanic$Sex, Titanic$AgeCat)

# Print the passengers by sex vs age

passengers\_by\_sex\_age

Child Adult

Female 30 152

Male 31 398