**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

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

**DAY 1 – LAB ASSESSMENT**

**Reg No:**

**Name:**

1. Write a R program to take input from the user (name and age) and display the values. Also print the version of R installation.

**Source code:**

name = readline(prompt="Input your name: ")

age = readline(prompt="Input your age: ")

print(paste("My name is",name, "and I am",age ,"years old."))

print(R.version.string)

OUTPUT:

Input your name:

Input your age:

[1] "My name is and I am years old."

[1] "R version 4.1.2 (2021-11-01)"

**2.** Write a R program to get the details of the objects in memory. 

SOURCE CODE:

name = "Python";

n1 = 10;

n2 = 0.5

nums = c(10, 20, 30, 40, 50, 60)

print(ls())

print("Details of the objects in memory:")

print(ls.str())

OUTPUT:

[1] "n1" "n2" "name" "nums"

[1] "Details of the objects in memory:"n1 : num 10

n2 : num 0.5

name : chr "Python"

nums : num [1:6] 10 20 30 40 50 60

1. Write a R program to create a sequence of numbers from 20 to 50 and find the mean of numbers from 20 to 60 and sum of numbers from 51 to 91.   
   SOURCE CODE:

print("Sequence of numbers from 20 to 50:")

print(seq(20,50))

print("Mean of numbers from 20 to 60:")

print(mean(20:60))

print("Sum of numbers from 51 to 91:")

print(sum(51:91))

OUTPUT:

[**1] "Sequence of numbers from 20 to 50:" [1] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44**

**[26] 45 46 47 48 49 50**

**[1] "Mean of numbers from 20 to 60:"[1] 40**

**[1] "Sum of numbers from 51 to 91:"[1] 2911**

1. Write a R program to create a vector which contains 10 random integer values between -50 and +50.   
   SOURCE CODE:

v = sample(-50:50, 10, replace=TRUE)

print("Content of the vector:")

print("10 random integer values between -50 and +50:")

print(v)

OUTPUT:

[1] "Content of the vector:"

[1] "10 random integer values between -50 and +50:"

[1] 35 -14 0 46 -49 4 23 -39 -31 30

**5.** Write a R program to get the first 10 Fibonacci numbers. 

SOURCE CODE:

Fibonacci <- numeric(10)

Fibonacci[1] <- Fibonacci[2] <- 1

for (i in 3:10) Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]

print("First 10 Fibonacci numbers:")

print(Fibonacci)

OUTPUT:

[1] "First 10 Fibonacci numbers:"

[1] 1 1 2 3 5 8 13 21 34 55

1. Write a R program to get all prime numbers up to a given number (based on the sieve of Eratosthenes).

SOURCE CODE:

prime\_numbers <- function(n) {

if (n >= 2) {

x = seq(2, n)

prime\_nums = c()

for (i in seq(2, n)) {

if (any(x == i)) {

prime\_nums = c(prime\_nums, i)

x = c(x[(x %% i) != 0], i)

}

}

return(prime\_nums)

}

else

{

stop("Input number should be at least 2.")

}

}

prime\_numbers(12)

OUTPUT:

[1] 2 3 5 7 11

**7.** Write a R program to print the numbers from 1 to 100 and print "Fizz" for multiples of 3, print "Buzz" for multiples of 5, and print "FizzBuzz" for multiples of both. 

SOURCE CODE:

for (n in 1:100) {

if (n %% 3 == 0 & n %% 5 == 0) {print("FizzBuzz")}

else if (n %% 3 == 0) {print("Fizz")}

else if (n %% 5 == 0) {print("Buzz")}

else print(n)

}

OUTPUT:

[1] 1

[1] 2[1] "Fizz"

[1] 4

[1] "Buzz"

[1] "Fizz"

[1] 7

[1] 8

[1] "Fizz"

[1] "Buzz"

[1] 11

[1] "Fizz"

[1] 13

[1] 14

[1] "FizzBuzz"

[1] 16

[1] 17

[1] "Fizz"

[1] 19

[1] "Buzz"[1] "Fizz"

[1] 22

[1] 23

[1] "Fizz"

[1] "Buzz"

[1] 26

[1] "Fizz"

[1] 28

[1] 29

[1] "FizzBuzz"

[1] 31

[1] 32

[1] "Fizz"

[1] 34

[1] "Buzz"

[1] "Fizz"[1] 37

[1] 38

[1] "Fizz"

[1] "Buzz"

[1] 41

[1] "Fizz"

[1] 43

[1] 44[1] "FizzBuzz"

[1] 46

[1] 47

[1] "Fizz"

[1] 49

[1] "Buzz"[1] "Fizz"

[1] 52

[1] 53

[1] "Fizz"

[1] "Buzz"

[1] 56

[1] "Fizz"[1] 58

[1] 59

[1] "FizzBuzz"

[1] 61

[1] 62

[1] "Fizz"

[1] 64

[1] "Buzz"

[1] "Fizz"[1] 67

[1] 68

[1] "Fizz"

[1] "Buzz"

[1] 71

[1] "Fizz"[1] 73

[1] 74

[1] "FizzBuzz"

[1] 76

[1] 77

[1] "Fizz"[1] 79

[1] "Buzz"

[1] "Fizz"

[1] 82

[1] 83

[1] "Fizz"[1] "Buzz"

[1] 86

[1] "Fizz"

[1] 88

[1] 89

[1] "FizzBuzz"

[1] 91

[1] 92[1] "Fizz"

[1] 94

[1] "Buzz"

[1] "Fizz"

[1] 97

[1] 98[1] "Fizz"

[1] "Buzz"

1. Write a R program to extract first 10 english letter in lower case and last 10 letters in upper case and extract letters between 22nd to 24th letters in upper case.

SOURCE CODE:

print("First 10 letters in lower case:")

t = head(letters, 10)

print(t)

print("Last 10 letters in upper case:")

t = tail(LETTERS, 10)

print(t)

print("Letters between 22nd to 24th letters in upper case:")

e = tail(LETTERS[22:24])

print(e)

OUTPUT:

[1] "First 10 letters in lower case:" [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"[1] "Last 10 letters in upper case:" [1] "Q" "R" "S" "T" "U" "V" "W" "X" "Y" "Z"[1] "Letters between 22nd to 24th letters in upper case:"[1] "V" "W" "X"

**9.** Write a R program to find the factors of a given number. 

SOURCE CODE:

**print\_factors = function(n) {**

**print(paste("The factors of",n,"are:"))**

**for(i in 1:n) {**

**if((n %% i) == 0) {**

**print(i)**

**}**

**}**

**}**

**print\_factors(4)**

**print\_factors(7)**

**print\_factors(12)**

**OUTPUT:**

[1] "The factors of 4 are:"

[1] 1[1] 2

[1] 4

[1] "The factors of 7 are:"

[1] 1

[1] 7

[1] "The factors of 12 are:"

[1] 1

[1] 2

[1] 3

[1] 4

[1] 6

[1] 12

1. Write a R program to find the maximum and the minimum value of a given vector.   
   SOURCE CODE:

nums = c(10, 20, 30, 40, 50, 60)

print('Original vector:')

print(nums)

print(paste("Maximum value of the said vector:",max(nums)))

print(paste("Minimum value of the said vector:",min(nums)))

OUTPUT:

[1] "Original vector:"

[1] 10 20 30 40 50 60[1] "Maximum value of the said vector: 60"

[1] "Minimum value of the said vector: 10"

**11.** Write a R program to get the unique elements of a given string and unique numbers of vector. 

SOURCE CODE:

str1 = "The quick brown fox jumps over the lazy dog."

print("Original vector(string)")

print(str1)

print("Unique elements of the said vector:")

print(unique(tolower(str1)))

nums = c(1, 2, 2, 3, 4, 4, 5, 6)

print("Original vector(number)")

print(nums)

print("Unique elements of the said vector:")

print(unique(nums))

OUTPUT:

[1] "Original vector(string)"

[1] "The quick brown fox jumps over the lazy dog."[1] "Unique elements of the said vector:"

[1] "the quick brown fox jumps over the lazy dog."

[1] "Original vector(number)"

[1] 1 2 2 3 4 4 5 6

[1] "Unique elements of the said vector:"

[1] 1 2 3 4 5 6

**12.** Write a R program to create three vectors a,b,c with 3 integers. Combine the three vectors to become a 3×3 matrix where each column represents a vector. Print the content of the matrix. 

SOURCE CODE:

a<-c(1,2,3)

b<-c(4,5,6)

c<-c(7,8,9)

m<-cbind(a,b,c)

print("Content of the said matrix:")

print(m)

OUTPUT:

[1] "Content of the said matrix:"

a b c

[1,] 1 4 7

[2,] 2 5 8

[3,] 3 6 9

1. Write a R program to create a list of random numbers in normal distribution and count occurrences of each value.

SOURCE CODE:

**n = floor(rnorm(1000, 50, 100))**

**print('List of random numbers in normal distribution:')**

**print(n)**

**t = table(n)**

**print("Count occurrences of each value:")**

**print(t)**

**OUTPUT:**

[1] "List of random numbers in normal distribution:" [1] -23 19 123 51 -115 13 30 96 73 255 46 28 44 34

[15] 91 -50 -74 50 208 -234 89 146 79 99 79 23 -52 146

[29] 113 -82 -21 49 43 -16 -146 41 173 -84 -139 -112 63 124

[43] 80 -88 -1 -27 -27 57 155 -136 -44 178 66 -6 9 -2

[57] -93 83 134 275 83 -179 132 32 74 -85 80 -91 -111 -9

[71] 135 196 -96 55 237 129 101 135 173 -263 74 158 -10 72

[85] 58 57 210 176 164 40 148 68 162 201 7 -9 295 11

[99] 140 191 69 133 102 42 104 146 9 122 9 43 168 -61

[113] 43 90 -42 88 19 38 59 66 198 -60 -6 160 -55 51

[127] -46 -74 -27 -100 55 -69 40 154 -6 228 -56 76 217 63

[141] 44 36 56 63 58 79 48 63 212 77 29 3 -43 136

[155] 160 13 -9 64 48 155 37 -9 -3 71 56 -77 176 51

[169] -49 208 98 -12 168 -79 134 11 196 94 -60 -130 147 123

[183] -169 38 -43 102 156 37 -14 -65 -2 37 133 -1 84 25

[197] 138 65 75 287 99 129 39 32 -66 137 131 96 88 44

[211] 55 -243 169 191 138 8 62 19 -39 195 94 -65 -46 146

[225] 363 34 136 -141 -26 41 -7 122 -119 112 10 32 27 72

[239] -17 217 50 -119 -110 69 -53 39 -49 117 61 -6 -8 161

[253] 123 -143 70 -30 195 -48 131 100 293 10 4 -41 210 186

[267] -145 25 -38 175 73 17 77 130 81 119 0 -64 -86 37

[281] 0 -39 26 95 -69 -119 128 43 228 -106 142 -84 128 28

[295] 94 51 9 198 56 82 91 90 -35 -1 130 250 -123 68

[309] 112 276 259 -22 -42 204 14 97 178 49 -7 223 73 159

[323] 103 150 154 167 278 82 141 -207 67 25 -12 72 -25 63

[337] 5 -21 148 169 84 130 11 -46 33 21 212 -19 108 91

[351] -168 84 307 20 85 41 -94 82 151 -90 26 41 123 -26

[365] 22 106 -23 -36 -6 -55 -44 212 85 3 32 52 -16 60

[379] -24 75 -9 12 155 113 -10 -218 183 22 130 -90 -39 -62

[393] 40 57 203 61 47 39 40 -159 76 3 -72 4 144 157

[407] 113 -124 10 68 -36 -123 80 -13 39 -33 18 19 147 104

[421] 138 -105 144 102 44 130 78 70 141 -15 170 -196 70 81

[435] 36 181 -146 46 -13 83 159 -11 -18 -71 67 171 70 49

[449] 79 220 84 -43 161 280 219 109 141 119 111 -66 1 40

[463] 72 75 74 46 13 27 94 161 49 61 138 41 75 108

[477] -15 27 -92 38 70 5 -118 5 55 -1 310 304 190 -89

[491] -70 77 121 76 55 10 143 29 -31 115 -5 25 -75 43

[505] 19 110 3 -2 -23 87 55 -38 -131 88 -111 74 89 126

[519] 51 -13 355 -19 4 71 131 -17 160 -92 168 61 99 126

[533] 155 142 48 65 124 69 26 145 -12 -143 29 289 11 135

[547] 55 -41 -3 -103 47 13 4 -93 5 139 86 75 63 151

[561] 73 124 131 89 18 56 57 127 113 255 121 168 74 223

[575] -18 200 48 10 184 16 169 -118 -6 -18 141 -8 67 -40

[589] -34 101 144 -145 -175 132 -10 7 197 48 95 -97 -140 -41

[603] 63 68 -11 -24 16 99 -52 -112 145 -66 98 -41 190 25

[617] -3 -36 11 -123 -20 -16 90 171 122 195 207 188 -8 150

[631] 116 -11 84 65 -99 -271 -39 -87 -10 -30 137 38 48 106

[645] -16 178 135 56 -23 100 111 -145 113 37 8 -43 80 26

[659] 103 32 -2 70 104 126 30 61 -114 106 -4 -70 -31 125

[673] -22 33 264 139 107 0 120 86 89 93 55 -18 23 20

[687] 82 141 139 42 116 -10 -65 -56 -101 137 125 31 -47 29

[701] 268 -78 198 190 157 -111 145 -23 367 23 34 26 156 56

[715] -59 141 209 243 -100 -22 36 -88 -10 102 43 82 -42 139

[729] 32 9 48 255 229 190 6 21 -63 196 186 177 -92 -241

[743] 55 83 -95 16 -89 -90 39 -43 39 168 -67 159 71 80

[757] 103 29 121 155 125 199 76 -35 205 -46 -34 -11 78 -51

[771] 257 182 228 -61 22 97 89 103 13 125 43 -86 148 135

[785] -31 130 -81 -16 67 122 283 107 78 -120 54 91 70 7

[799] 165 49 -25 69 30 -22 63 -23 1 151 78 214 -149 107

[813] 212 147 -163 66 105 -100 -179 59 -226 123 55 275 131 160

[827] 33 -100 171 13 -20 60 141 66 335 50 50 161 4 146

[841] -58 135 -123 171 120 106 102 -201 19 70 19 54 -28 -163

[855] 127 -62 91 224 -43 142 -173 23 57 163 124 39 108 123

[869] -6 -184 83 14 14 -60 -55 -102 18 97 65 10 188 78

[883] 149 10 79 -13 -16 -256 27 -2 -139 -109 -23 -75 104 110

[897] 122 -27 56 84 -22 63 -38 -66 -45 7 -66 264 158 130

[911] 34 -5 59 -60 194 124 -109 71 156 -85 -117 -112 74 -99

[925] 62 -109 193 173 84 -47 96 88 42 83 146 123 37 -92

[939] -13 168 107 225 -160 49 188 -16 153 -62 34 -141 9 81

[953] 114 80 85 189 10 258 5 5 21 -53 55 182 259 41

[967] 124 236 -70 87 26 116 -51 129 43 -122 2 80 122 -222

[981] -94 120 203 -129 166 64 17 -70 -117 171 110 182 42 49

[995] 150 -76 210 -8 125 3[1] "Count occurrences of each value:"n-271 -263 -256 -243 -241 -234 -226 -222 -218 -207 -201 -196 -184 -179 -175 -173

1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1

-169 -168 -163 -160 -159 -149 -146 -145 -143 -141 -140 -139 -136 -131 -130 -129

1 1 2 1 1 1 2 3 2 2 1 2 1 1 1 1

-124 -123 -122 -120 -119 -118 -117 -115 -114 -112 -111 -110 -109 -106 -105 -103

1 4 1 1 3 2 2 1 1 3 3 1 3 1 1 1

-102 -101 -100 -99 -97 -96 -95 -94 -93 -92 -91 -90 -89 -88 -87 -86

1 1 4 2 1 1 1 2 2 4 1 3 2 2 1 2

-85 -84 -82 -81 -79 -78 -77 -76 -75 -74 -72 -71 -70 -69 -67 -66

2 2 1 1 1 1 1 1 2 2 1 1 4 2 1 5

-65 -64 -63 -62 -61 -60 -59 -58 -56 -55 -53 -52 -51 -50 -49 -48

3 1 1 3 2 4 1 1 2 3 2 2 2 1 2 1

-47 -46 -45 -44 -43 -42 -41 -40 -39 -38 -36 -35 -34 -33 -31 -30

2 4 1 2 6 3 4 1 4 3 3 2 2 1 3 2

-28 -27 -26 -25 -24 -23 -22 -21 -20 -19 -18 -17 -16 -15 -14 -13

1 4 2 2 2 7 5 2 2 2 4 2 7 2 1 5

-12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3

3 4 6 5 4 2 7 2 1 3 5 4 3 2 1 5

4 5 6 7 8 9 10 11 12 13 14 16 17 18 19 20

5 6 1 4 2 6 8 5 1 6 3 3 2 3 7 2

21 22 23 25 26 27 28 29 30 31 32 33 34 36 37 38

3 3 4 5 6 4 2 5 3 1 6 3 5 3 6 4

39 40 41 42 43 44 46 47 48 49 50 51 52 54 55 56

7 5 6 4 8 4 3 2 7 7 4 5 1 2 11 7

57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72

5 2 3 2 5 2 9 2 4 4 4 4 4 8 4 4

73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88

4 6 5 4 3 5 5 7 3 5 6 7 3 2 2 4

89 90 91 93 94 95 96 97 98 99 100 101 102 103 104 105

5 3 5 1 4 2 3 3 2 4 2 2 5 4 4 1

106 107 108 109 110 111 112 113 114 115 116 117 119 120 121 122

4 4 3 1 3 2 2 5 1 1 3 1 2 3 3 6

123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138

7 6 5 3 2 2 3 7 5 2 2 2 6 2 3 4

139 140 141 142 143 144 145 146 147 148 149 150 151 153 154 155

4 1 7 3 1 3 3 6 3 3 1 3 3 1 2 5

156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171

3 2 2 3 4 4 1 1 1 1 1 1 6 3 1 5

173 175 176 177 178 181 182 183 184 186 188 189 190 191 193 194

3 1 2 1 3 1 3 1 1 2 3 1 4 2 1 1

195 196 197 198 199 200 201 203 204 205 207 208 209 210 212 214

3 3 1 3 1 1 1 2 1 1 1 2 1 3 4 1

217 219 220 223 224 225 228 229 236 237 243 250 255 257 258 259

2 1 1 2 1 1 3 1 1 1 1 1 3 1 1 2

264 268 275 276 278 280 283 287 289 293 295 304 307 310 335 355

2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1

363 367

1 1

**14.** Write a R program to create three vectors numeric data, character data and logical data. Display the content of the vectors and their type. 

*SOURCE CODE:*

*a = c(1, 2, 5, 3, 4, 0, -1, -3)*

*b = c("Red", "Green", "White")*

*c = c(TRUE, TRUE, TRUE, FALSE, TRUE, FALSE)*

*print(a)*

*print(typeof(a))*

*print(b)*

*print(typeof(b))*

*print(c)*

*print(typeof(c))*

OUTPUT:

[1] 1 2 5 3 4 0 -1 -3[1] "double"

[1] "Red" "Green" "White"

[1] "character"

[1] TRUE TRUE TRUE FALSE TRUE FALSE

[1] "logical"

1. Write a R program to create a 5 x 4 matrix , 3 x 3 matrix with labels and fill the matrix by rows and 2 × 2 matrix with labels and fill the matrix by columns.

SOURCE CODE:

**m1 = matrix(1:20, nrow=5, ncol=4)**

**print("5 × 4 matrix:")**

**print(m1)**

**cells = c(1,3,5,7,8,9,11,12,14)**

**rnames = c("Row1", "Row2", "Row3")**

**cnames = c("Col1", "Col2", "Col3")**

**m2 = matrix(cells, nrow=3, ncol=3, byrow=TRUE, dimnames=list(rnames, cnames))**

**print("3 × 3 matrix with labels, filled by rows: ")**

**print(m2)**

**print("3 × 3 matrix with labels, filled by columns: ")**

**m3 = matrix(cells, nrow=3, ncol=3, byrow=FALSE, dimnames=list(rnames, cnames))**

**print(m3)**

**OUTPUT:**

**.**

[1] "5 × 4 matrix:" [,1] [,2] [,3] [,4]

[1,] 1 6 11 16

[2,] 2 7 12 17

[3,] 3 8 13 18

[4,] 4 9 14 19

[5,] 5 10 15 20

[1] "3 × 3 matrix with labels, filled by rows: "

Col1 Col2 Col3

Row1 1 3 5

Row2 7 8 9

Row3 11 12 14

[1] "3 × 3 matrix with labels, filled by columns: "

Col1 Col2 Col3

Row1 1 7 11

Row2 3 8 12

Row3 5 9 14

**16.** Write a R program to create an array, passing in a vector of values and a vector of dimensions. Also provide names for each dimension. 

SOURCE CODE:

a = array(

6:30,

dim = c(4, 3, 2),

dimnames = list(

c("Col1", "Col2", "Col3", "Col4"),

c("Row1", "Row2", "Row3"),

c("Part1", "Part2")

)

)

print(a)

OUTPUT:

, , Part1

Row1 Row2 Row3

Col1 6 10 14

Col2 7 11 15

Col3 8 12 16

Col4 9 13 17

, , Part2

Row1 Row2 Row3

Col1 18 22 26

Col2 19 23 27

Col3 20 24 28

Col4 21 25 29

**17.** Write a R program to create an array with three columns, three rows, and two "tables", taking two  vectors as input to the array.  Print the array. 

SOURCE CODE

v1 = c(1, 3, 5, 7)

v2 = c(2, 4, 6, 8, 10)

arra1 = array(c(v1, v2),dim = c(3,3,2))

print(arra1)

:

OUTPUT:

, , 1

[,1] [,2] [,3]

[1,] 1 7 6

[2,] 3 2 8

[3,] 5 4 10

, , 2

[,1] [,2] [,3]

[1,] 1 7 6

[2,] 3 2 8

[3,] 5 4 10

1. Write a R program to create a list of elements using vectors, matrices and a function. Print the content of the list.

SOURCE CODE:

l = list(

c(1, 2, 2, 5, 7, 12),

month.abb,

matrix(c(3, -8, 1, -3), nrow = 2),

asin

)

print("Content of the list:")

print(l)

OUTPUT:

[1] "Content of the list:"[[1]]

[1] 1 2 2 5 7 12

[[2]]

[1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"

[[3]]

[,1] [,2]

[1,] 3 1

[2,] -8 -3

[[4]]

function (x) .Primitive("asin")