NARAYANAN.R

191911569

ITA04 STATISTICS WITH R PROGRAMMING

ASSESSMENT DAY 2

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.

SOLUTION:

```
name <- readline(prompt = "Enter your name: ")
age <- readline(prompt = "Enter your age: ")
cat("R version:", R.version.string, "\n")
```

OUTPUT

```
Enter your name: NAVEEN
Enter your age: 21
R version: R version 4.1.2 (2021-11-01)
```

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

SOLUTION:

```
name = "REACT JS";
```

```
n1 =  5;
n2 =  2
nums = c(10, 200, 30, 400, 500, 600)
print(ls())
print("Details OF objects in memory:")
print(ls.str())
```

OUTPUT

```
[1] "n1"   "n2"   "name" "nums"
[1] "Details OF objects in memory:"
n1 : num 5
n2 : num 2
name : chr "REACT JS"
nums : num [1:6] 10 200 30 400 500 600
```

3. Write a R program to create a sequence of numbers from 10 to 50 and find the

mean of numbers from 35 to 56 and sum of numbers from 400 to 700.

SOLUTION

```
print("Sequence of numbers from 10 to 50:")
print(seq(10,50))
print("Mean of numbers from 35 to 56:")
```

```
print(mean(250:260))

print("Sum of numbers from 400 to 700:")

print(sum(400:700))
```

OUTPUT:

```
[1] "Sequence of numbers from 10 to 50:"

[1] 10 11 12 13 14 15 16 17 18 1920 21 22 23 24 25 26 27 28 29
30 31 32 33 34

[26] 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50

[1] "Mean of numbers from 35 to 56:"

[1] 255

[1] "Sum of numbers from 400 to 700:"

[1] 165550
```

4. Write a R program to create a vector which contains 10 random integer values

between -50 and +50.

SOLUTION:

```
v=sample(-300:300, 10, replace=TRUE)

print("Content of the vector:")

print("10 random integer values between -300 and +300:")

print(v)
```

OUPUT:

```
[1]"Content of the vector:"[1] "10 random integer values
between -300 and +300:"

[1]  211  137  205 -266  -25  219 -109  268 -137  231
```

5. Write a R program to get all prime numbers up to a given number (based on

the sieve of Eratosthenes).

SOLUTION:

```
prime_nO <- 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_nO(100)
```

OUTPUT:

[1] 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

6. Write a R program to extract first 10 english letter in lower case and last 10

letters in upper case and extract letters between 22 nd to 24 th letters in upper case.

SOLUTION:

```
print("First 3 letters in lower case:")
H = head(letters, 3)
print(H)
print("Last 11 letters in upper case:")
H = tail(LETTERS, 11)
print(H)
print("Letters between 21nd to 26th letters in upper case:")
K = tail(LETTERS[21:26])
print(K)
```

OUTPUT:

```
[1] "First 3 letters in lower case:"
[1] "a" "b" "c"
[1] "Last 11 letters in upper case:"
[1] "P" "Q" "R" "S" "T" "U" "V" "W" "X" "Y" "Z"
[1] "Letters between 21nd to 26th letters in upper case:"
[1] "U" "V" "W" "X" "Y" "Z"
```

7. Write a R program to find the maximum and the minimum value of a given

vector.

SOLUTION;

```
nums = c(110, 210, 310, 410, 510, 610)
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] 110 210 310 410 510 610
[1] "Maximum value of the said vector: 610"
[1] "Minimum value of the said vector: 110"
```

8. Write a R program to get the unique elements of a given string and unique

numbers of vector.

SOLUTION;

```
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(10, 20, 20, 30, 40, 40, 50, 60)
```

```
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]10 20 20 30 40 4050 60
[1] "Unique elements of the said vector:"
[1] 10 20 30 40 50 60
```

9. 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.

SOLUTION:

```
h<-c(5,4,3)
i<-c(1,5,6)
j<-c(7,8,10)
F<-cbind(h,i,j)
```

```
print("Content of the said matrix:")
print(F)
```

OUTPUT:

```
[1] "Content of the said matrix:"
h i  j
[1,] 5 1  7
[2,] 4 5  8
[3,] 3 6 10
```

10. Write a R program to create a list of random numbers in normal distribution

and count occurrences of each value.

SOLUTION:

```
n = floor(rnorm(10, 50, 20))
print('List of random numbers in normal distribution:')
print(n)
L= table(n)
print("Count occurrences of each value:")
print(L)
```

OUTPUT:

```
[1] "List of random numbers in normal distribution:"
 [1]49 50 92 20 36 42 64 34 63 33
[1] "Count occurrences of each value:"
n
20 33 34 36 42 49 50 63 64 92
```

1 1 1 1 1 1 1 1 1 1

11. Write a R program to create three vectors numeric data, character data and

logical data. Display the content of the vectors and their type.

SOLUTION

```
a = c(0, 2, 6, 5,3, 0, -1, -4)
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]  0  2  65  3  0 -1 -4
[1] "double"
[1] "Red"   "Green" "White"
[1] "character"
[1]  TRUE  TRUE  TRUE FALSE  TRUE FALSE
[1] "logical"
```

12. 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.

SOLUTION:

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

print("5 × 4 matrix:")

print(m1)

cells = c(1,13,5,9,8,19,11,12,14)

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 \303\227 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 \303\227 3 matrix with labels, filled by rows: "
     Col1 Col2 Col3
Row1    1   13    5
Row2    9    8   19
Row3   11   12   14
[1] "3 \303\227 3 matrix with labels, filled by columns: "
     Col1 Col2 Col3
Row1    1    9   11
Row2   13    8   12
Row3    5   19   14
```

13. 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.

SOLUTION:

```
a =  array(
  6:20,
  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    7   11
Col2   19    8   12
Col3   20    9   13
Col4    6   10   14
```

14. 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.

SOLUTION:

```
v1 = c(1, 3, 7, 9)
```

```
v2 = c(12, 14, 16, 18, 10)
arra1 = array(c(v1, v2),dim = c(3,3,2))
print(arra1)
```

OUTPUT:

```
[,1] [,2] [,3]
[1,]    1    9   16
[2,]    3   12   18
[3,]    7   14   10

, , 2

     [,1] [,2] [,3]
[1,]    1    9   16
[2,]    3   12   18
[3,]    7   14   10
```

15. Write a R program to create a list of elements using vectors, matrices and a

functions. Print the content of the list.

SOLUTION:

```
l = list(
  c(1, 1, 12, 15, 7, 12),
```

```
  month.abb,
  matrix(c(5, -8, 6, -3), nrow = 2),
  asin
)
print("Content of the list:")
print(l)
```

OUTPUT:

```
[1] "Content of the list:"
[[1]]
[1]  1  1 12 15  7 12


[[2]]
 [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep"
"Oct" "Nov" "Dec"


[[3]]
     [,1] [,2]
[1,]    5    6
[2,]   -8   -3


[[4]]
function (x)  .Primitive("asin")
```