Practice Questions (Short Read) in R - Programming for Data Science

Read: 30min

(Ashwini Mathur (CSSP) JAIN - SET, BANGALORE)

Note: Read all the Questions very carefully.(Kindly Implement the small tasks with the Help of R - Programming for better Understanding the basic concepts)

###

Explain what is R?

R is data analysis software which is used by analysts, quants, statisticians, data scientists and others.

List out some of the functions that R provides?

The function that R provides are

- Mean
- Median
- Distribution
- Covariance
- Regression
- Non-linear

What is the function used for adding datasets in R?

rbind function can be used to join two data frames (datasets). The two data frames must have the same variables, but they do not have to be in the same order.

In R how missing values are represented ?

In R missing values are represented by NA (Not Available), why impossible values are represented by the symbol NaN (not a number).

What are the different data structures in R? Briefly explain about them.

Data Structure	Description
Vector	A vector is a sequence of data elements of the same basic type. Members in a vector are called components.
List	Lists are the R objects which contain elements of different types like − numbers, strings, vectors or another list inside it.
Matrix	A matrix is a two-dimensional data structure. Matrices are used to bind vectors from the same length. All the elements of a matrix must be of the same type (numeric, logical, character, complex).
Dataframe	A data frame is more generic than a matrix, i.e different columns can have different data types (numeric, character, logical, etc). It combines features of matrices and lists like a rectangular list.

How can you load a .csv file in R?

- Loading a .csv file in R is quite easy.
- All you need to do is use the “read.csv()” function and specify the path of the file.

```
house<-read.csv("C:/Users/John/Desktop/house.csv")
```

How do you install a package in R?

The below command is used to install a package in R:

```
install.packages(“<package_name>”)
```

How would you write a custom function in R? Give an example.

This is the syntax to write a custom function In R:

<object-name>=function(x){

}

Let’s look at an example to create a custom function in R ->

```
fun1<-function(x){ ifelse(x>5,100,0) }
v<-c(1,2,3,4,5,6,7,8,9,10)
fun1(v)->v
```

Name some functions available in “dplyr” package.

Functions in dplyr package:

- filter
- select
- mutate
- arrange
- Count

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.

Sample Solution :

R Programming 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))
```

Sample 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
```

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

Sample Solution :

R Programming 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)
```

Sample Output:

```
[1] "First 10 Fibonacci numbers:"

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

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

Sample Solution :

R Programming Code :

```
v = sample(-50:50, 10, replace=TRUE) print("Content of the
vector:") print("10 random integer values between -50 and
+50:") print(v)
```

Sample Output:

```
[1] "Content of the vector:"
[1] "10 random integer values between -50 and +50:"
[1]  31 -13 -21  42  49 -39  20  12  39  -2
```

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.

Note: Use built-in datasets letters and LETTERS.

Sample Solution :

R Programming 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)
```

Sample 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"
```

What is advantage of using apply family of functions in R?

The apply function allows us to make entry-by-entry changes to data frames and matrices.

The usage in R is as follows:

apply(X, MARGIN, FUN, …)

where:

X is an array or matrix;

MARGIN is a variable that determines whether the function is applied over rows (MARGIN=1), columns (MARGIN=2), or both (MARGIN=c(1,2));

FUN is the function to be applied.

If MARGIN=1, the function accepts each row of X as a vector argument, and returns a vector of the results. Similarly, if MARGIN=2 the function acts on the columns of X. Most impressively, when MARGIN=c(1,2) the function is applied to every entry of X.

Advantage:

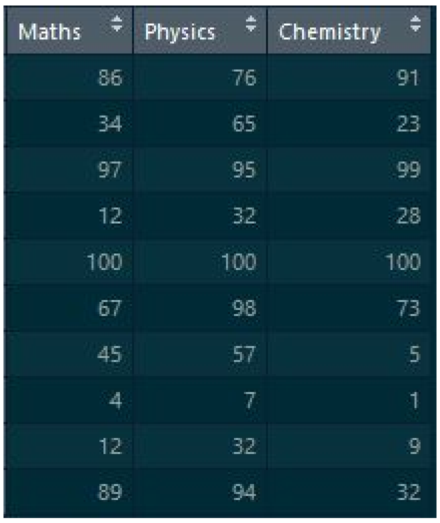
With the apply function we can edit every entry of a data frame with a single line command. No auto-filling, no wasted CPU cycles.

Give examples of “rbind()” and “cbind()” functions in R

Cbind(): As the name suggests, it is used to bind two columns together. One fact to be kept in mind while binding two columns is, the number of rows in both the columns need to be same.

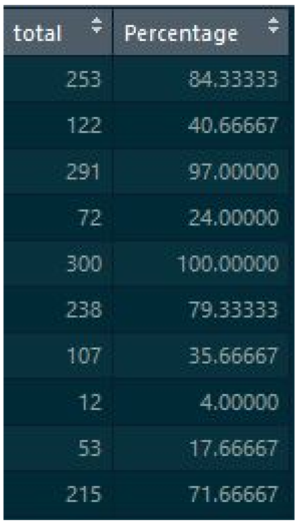
Let’s understand this with an example:

This is “Marks” data-set which comprises of marks in three subjects->



Maths	Physics	Chemistry
86	76	91
34	65	23
97	95	99
12	32	28
100	100	100
67	98	73
45	57	5
4	7	1
12	32	9
89	94	32

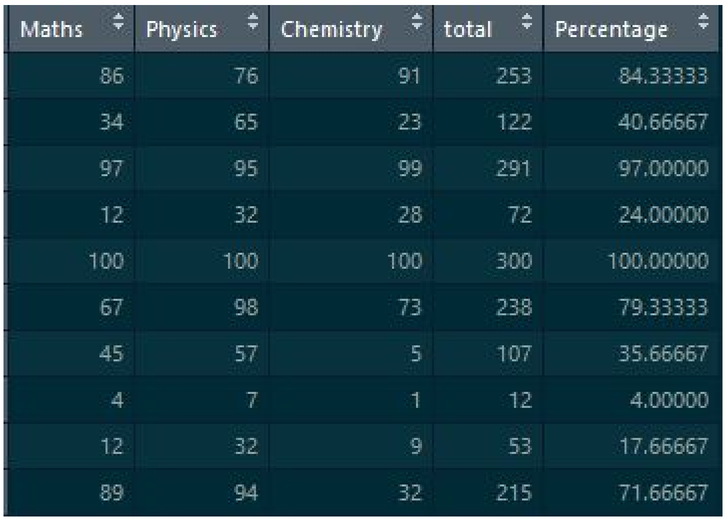
We’ll bind this with a new dataset “Percentage” which consists of two columns :-> “Total” and “Percentage”



total	Percentage
253	84.33333
122	40.66667
291	97.00000
72	24.00000
300	100.00000
238	79.33333
107	35.66667
12	4.00000
53	17.66667
215	71.66667

Let’s combine the columns from these two data-sets using the “cbind()” function->

```
cbind(Marks,Percentage)
```



Maths	Physics	Chemistry	total	Percentage
86	76	91	253	84.33333
34	65	23	122	40.66667
97	95	99	291	97.00000
12	32	28	72	24.00000
100	100	100	300	100.00000
67	98	73	238	79.33333
45	57	5	107	35.66667
4	7	1	12	4.00000
12	32	9	53	17.66667
89	94	32	215	71.66667

Since, the number of rows in both the data-sets is same we have combined the columns with the help of "cbind()" function

----

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.

Sample Solution :

R Programming Code :

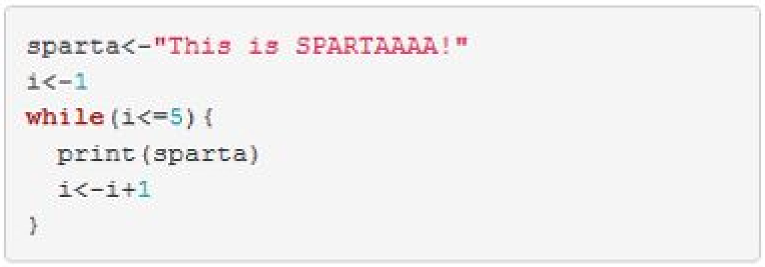
```
a<-c(1,2,3) b<-c(4,5,6) c<-c(7,8,9) m<-cbind(a,b,c)
print("Content of the said matrix:") print(m)
```

Sample Output:

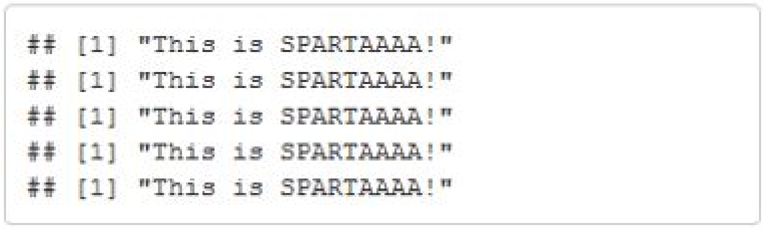
```
[1] "Content of the said matrix:"
   a b c
[1,] 1 4 7
[2,] 2 5 8
[3,] 3 6 9
```

Give examples of while and for loop in R.

While loop:

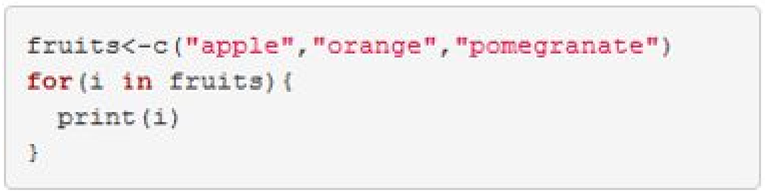


```r
sparta<-"This is SPARTAAAA!"
i<-1
while(i<=5){
  print(sparta)
  i<-i+1
}
```

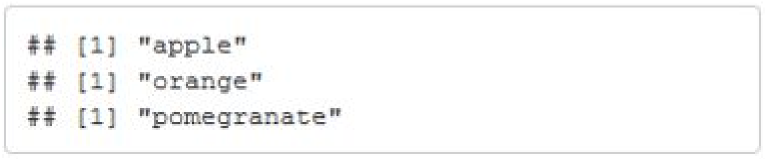


```
## [1] "This is SPARTAAAA!"
## [1] "This is SPARTAAAA!"
## [1] "This is SPARTAAAA!"
## [1] "This is SPARTAAAA!"
## [1] "This is SPARTAAAA!"
```

For loop:



```r
fruits<-c("apple","orange","pomegranate")
for(i in fruits){
  print(i)
}
```



```
## [1] "apple"
## [1] "orange"
## [1] "pomegranate"
```

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

Sample Solution :

R Programming 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))
```

Sample 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"
```

What are the different data objects in R?

There are 6 data objects in R. They are vectors, lists, arrays, matrices, data frames and tables.

R Programming 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)
```

Copy

Sample 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
```

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.

Sample Solution :

R Programming 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)
```

Sample 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
```

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.

Sample Solution :

R Programming 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)
```

Sample 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
```

What is the main difference between an Array and a matrix?

A matrix is always two dimensional as it has only rows and columns. But an array can be of any number of dimensions and each dimension is a matrix. For example a 3x3x2 array represents 2 matrices each of dimension 3x3.

What is R Base package?

This is the package which is loaded by default when R environment is set. It provides the basic functionalities like input/output, arithmetic calculations etc. in the R environment.

Write a R program to compute sum, mean and product of a given vector elements.

Sample Solution :

R Programming Code :

```
nums = c(10, 20, 30) print('Original vector:') print(nums)
print(paste("Sum of vector elements:",sum(nums)))
print(paste("Mean of vector elements:",mean(nums)))
print(paste("Product of vector elements:",prod(nums)))
```

Sample Output:

```
[1] "Original vector:"

[1] 10 20 30

[1] "Sum of vector elements: 60"

[1] "Mean of vector elements: 20"

[1] "Product of vector elements: 6000"
```

How R is used in logistic regression?

Logistic regression deals with measuring the probability of a binary response variable. In R the function glm() is used to create the logistic regression.

Write a R program to create a Dataframes which contain details of 5 employees and display the details.

Sample Solution :

R Programming Code :

```
Employees = data.frame(Name=c("Anastasia S","Dima R","Katherine S", "JAMES A","LAURA MARTIN"), Gender=c("M","M","F","F","M"), Age=c(23,22,25,26,32), Designation=c("Clerk","Manager","Exective","CEO","ASSISTANT"), SSN=c("123-34-2346","123-44-779","556-24-433","123-98-987","679-77-576") ) print("Details of the employees:") print(Employees)
```

Copy

Sample Output:

```
[1] "Details of the employees:"

        Name Gender Age Designation        SSN
```

```
1  Anastasia S      M  23      Clerk 123-34-2346
2       Dima R      M  22    Manager  123-44-779
3  Katherine S      F  25    Exective  556-24-433
4      JAMES A      F  26        CEO  123-98-987
5 LAURA MARTIN      M  32   ASSISTANT  679-77-576
```

How do you access the element in the 2nd column and 4th row of a matrix named M?

The expression M[4,2] gives the element at 4th row and 2nd column.

Explain general format of Matrices in R?

General format is

```
Mymatrix< - matrix (vector, nrow=r , ncol=c , byrow=FALSE,
dimnames = list ( char_vector_ rowname, char_vector_colnames))
```

Write a R program to create a data frame from four given vectors.

Sample Solution:

R Programming Code:

```
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
data frame:") print(name) print(score) print(attempts)
```

```
print(qualify) df = data.frame(name, score, attempts, qualify)
print(df)
```

Sample Output:

```
[1] "Original data frame:"
 [1] "Anastasia" "Dima"      "Katherine" "James"     "Emily"     "Michael"
 [7] "Matthew"   "Laura"     "Kevin"     "Jonas"
 [1] 12.5  9.0 16.5 12.0  9.0 20.0 14.5 13.5  8.0 19.0
 [1] 1 3 2 3 2 3 1 1 2 1
 [1] "yes" "no"  "yes" "no"  "no"  "yes" "yes" "no"  "no"  "yes"
        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
```

Write a R program to get the structure of a given data frame.

Sample Solution :

R Programming Code :

```
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:")
```

```
print(exam_data) print("Structure of the said data frame:")
print(str(exam_data))
```

Sample Output:

```
[1] "Original dataframe:"
        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
[1] "Structure of the said data frame:"
'data.frame':  10 obs. of  4 variables:
 $ name    : Factor w/ 10 levels "Anastasia","Dima",..: 1 2 6 4 3 10 9 8 7 5
 $ score   : num  12.5 9 16.5 12 9 20 14.5 13.5 8 19
 $ attempts: num  1 3 2 3 2 3 1 1 2 1
 $ qualify : Factor w/ 2 levels "no","yes": 2 1 2 1 1 2 2 1 1 2
NULL
```