R PROGRAMMING ASSIGNMENT-3

1.Write a program to concatenate 2 Data Frames?

ANS.

 # Data frame 1

```
df1 = data.frame(StudentId = c(101:106),
                 Product = c("Hindi", "English",
                             "Maths", "Science",
                             "Political Science",
                             "Physics"))
df1
```

Output:

	StudentId	Product
1	101	Hindi
2	102	English
3	103	Maths
4	104	Science
5	105	Political Science
6	106	Physics

 # Data frame 2

```
df2 = data.frame(StudentId = c(102, 104, 106,
                               107, 108),
                 State = c("Manglore", "Mysore",
```

```
                    "Pune", "Dehradun", "Delhi"))
df2
```

Output:

```
StudentId   State
1       102 Manglore
2       104   Mysore
3       106     Pune
4       107 Dehradun
5       108    Delhi
```

 # CONATENATING TWO DATA FRAMES

```
df = merge(x = df1, y = df2, by = "StudentId",
                          all.x = TRUE)
df
```

Output:

```
  StudentId          Product    State
1       101            Hindi     NA
2       102          English   Manglore
3       103            Maths     NA
4       104          Science   Mysore
5       105 Political Science     NA
6       106          Physics    Pune
```

2.WRITE A PROGRAM TO DROP ROW BY NAME?

ANS.

 # Data frame

```
df1 = data.frame(Name = c('George','Andrea',
'Micheal','Maggie','Ravi','Xien','Jalpa'),

                 Grade_score=c(4,6,2,9,5,7,8),

                 Mathematics1_score=c(45,78,44,89,66,49,72),

                 Science_score=c(56,52,45,88,33,90,47))
```

df1

Output:

So the resultant dataframe will be

	Name	grade_score	Mathematics1_score	science_score
1	George	4	45	56
2	Andrea	6	78	52
3	Micheal	2	44	45
4	Maggie	9	89	88
5	Ravi	5	66	33
6	Xien	7	49	90
7	Jalpa	8	72	47

Drop Row by row name :

```
### drop rows  using slice() function in R

Library(dplyr)

df1[!(row.name(df1)  %in%   c('1','2')),  ]
```

Output:

	Name	grade_score	Mathematics1_score	science_score
3	Micheal	2	44	45
4	Maggie	9	89	88
5	Ravi	5	66	33
6	Xien	7	49	90
7	Jalpa	8	72	47

3. Write a program to drop row by number?

ANS:

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)

exam_data <- exam_data[-c(2, 4, 6), ]

print(exam_data)
```

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
```

```
  name score attempts qualify
1  Anastasia  12.5        1     yes
3  Katherine  16.5        2     yes
5      Emily   9.0        2      no
7    Matthew  14.5        1     yes
8      Laura  13.5        1      no
9      Kevin   8.0        2      no
10     Jonas  19.0        1     yes
```

4. What is the significance of R programming?

ANS:

R plays a very important role in Data Science, you will be benefited with following operations in R.

- **You can run your code without any compiler** – R is an interpreted language. Hence we can run code without any compiler. R interprets the code and makes the development of code easier.
- **Many calculations done with vectors** – R is a vector language, so anyone can add functions to a single Vector without putting in a loop. Hence, R is powerful and faster than other languages.
- **Statistical Language** – R used in biology, genetics as well as in statistics. R is a turning complete language where any type of task can perform.

Reasons to become a user:

One of the most powerful characteristics of R is that it is open-source, meaning anyone can access the underlying code used to run the program and add their own code for free. This means that R:

1. will always be able to perform the newest statistical analyses as soon as anyone thinks of them;
2. will fix its bugs quickly and transparently; and
3. has brought together a community of programming and stats nerds (a.k.a., useRs) that you can turn to for help.

Anyone can write their own R code, which means anyone can add to the huge list of R's tools. Programmers submit their code to R in the form of

"packages." Some packages specialize in specific kinds of analyses, while other packages are much broader. For example, the "pwr" package by Stephane Champely specializes in conducting power analyses. In contrast, the "psych" package by APS Fellow William R. Revelle can do anything from descriptive statistics to item-response theory to mediation analyses. At the start of 2017, there are just under 10,000 packages available. And as soon as a new statistical approach is developed, someone will create a new package or add new tools to an

A final reason you should become a useR is that R is increasingly being used as an industry standard in the realm of data analytics, also known as "data science." Many companies (e.g., Facebook, Merck, Pfizer) that hire psychology PhD students recruit candidates who have a solid grasp of both statistics and programming. Learning R will make you a more attractive candidate if you apply for non academic jobs, and teaching R will provide your students with more career options.

Pros of R Language:

- R is the most comprehensive statistical analysis package, as new technology and ideas often appear first in R.
- R is an open-source that's why you can run R anywhere any time, and even sell it under conditions of the license.
- It is cross-platform which runs on many operating systems. It's best for GNU/Linux and Microsoft Windows.
- In R, everyone is welcomed to *provide bug fixes, code enhancements, and new packages.*

Cons of R Language:

- The quality of some packages in R is less than perfect.
- There's no customer support of R Language whom you can complain if something doesn't work.

5.Write an R program to change a column name?

ANS:

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, NA, 2, NA, 2, NA, 1, NA, 2, 1),
qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)
print("Original dataframe:")
print(exam_data)
print("Change column-name 'name' to 'student_name' of the said dataframe:")
colnames(exam_data)[which(names(exam_data) == "name")] = "student_name"
print(exam_data)
```

Output:

```
[1] "Original dataframe:"
        name score attempts qualify
1  Anastasia  12.5        1     yes
2       Dima   9.0       NA      no
3  Katherine  16.5        2     yes
```

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

[1] "Change column-name 'name' to 'student_name' of the said dataframe:"

student_name score attempts qualify

1 Anastasia 12.5 1 yes

2 Dima 9.0 NA no

3 Katherine 16.5 2 yes

4 James 12.0 NA no

5 Emily 9.0 2 no

6 Michael 20.0 NA yes

7 Matthew 14.5 1 yes

8 Laura 13.5 NA no

9 Kevin 8.0 2 no

10 Jonas 19.0 1 yes

6.Write a program to change more than one column name?

ANS:

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, NA, 2, NA, 2, NA, 1, NA, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

print("Original dataframe:")

print(exam_data)

print("Change more than one column name  of the said dataframe:")

colnames(exam_data)[which(names(exam_data) == "name")] = "student_name"

colnames(exam_data)[which(names(exam_data) == "score")] = "avg_score"

print(exam_data)
```

OUTPUT:

```
[1] "Original dataframe:"
       name score attempts qualify
1  Anastasia  12.5        1     yes
2       Dima   9.0       NA      no
3  Katherine  16.5        2     yes
4      James  12.0       NA      no
5      Emily   9.0        2      no
6    Michael  20.0       NA     yes
7    Matthew  14.5        1     yes
8      Laura  13.5       NA      no
9      Kevin   8.0        2      no
10     Jonas  19.0        1     yes
[1] "Change more than one column name  of the said dataframe:"
   student_name avg_score attempts qualify
1     Anastasia      12.5        1     yes
2          Dima       9.0       NA      no
3     Katherine      16.5        2     yes
4         James      12.0       NA      no
```

5	Emily	9.0	2	no
6	Michael	20.0	NA	yes
7	Matthew	14.5	1	yes
8	Laura	13.5	NA	no
9	Kevin	8.0	2	no
10	Jonas	19.0	1	yes