

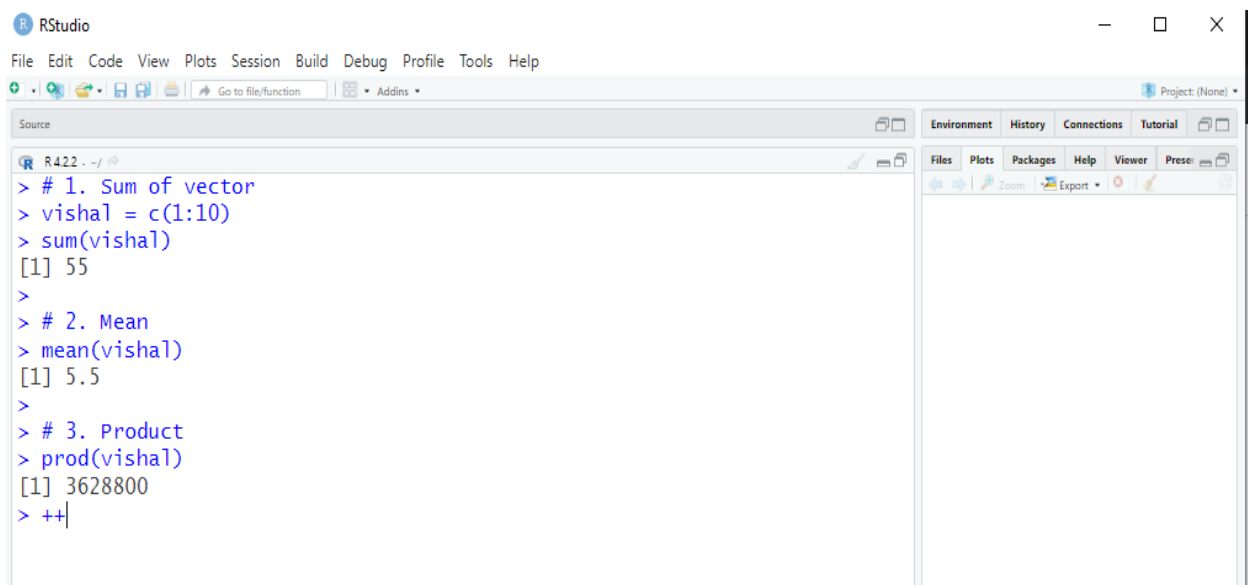
Big Data and Analytical Lab

Lab Assignment – 04

(BCSE0183)

Additional Problems on Basics of R Programming Language

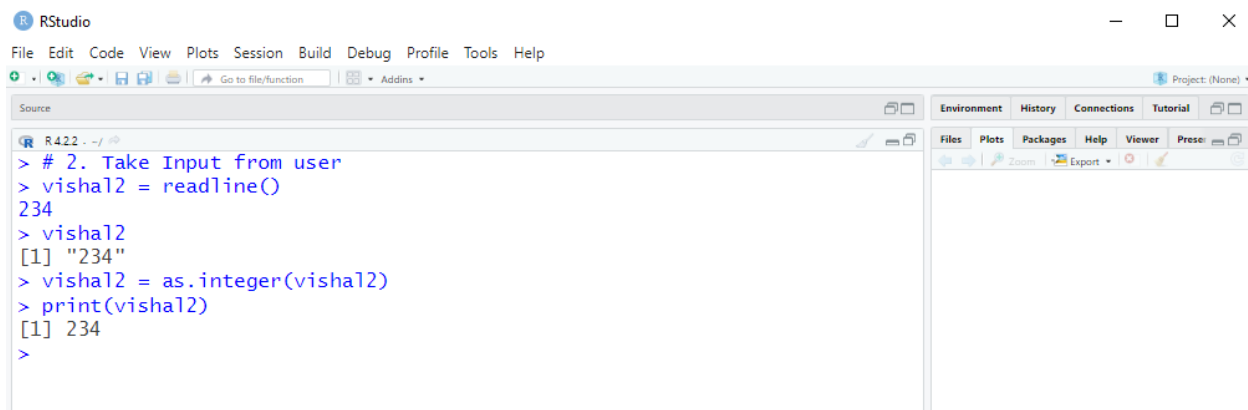
1) Write R Program to Find Sum, Mean and Product of Vector.



The screenshot shows the RStudio interface with a script editor on the left and the Environment, History, Connections, and Tutorial panels on the right. The script in the editor is as follows:

```
R 4.2.2 - ~/
> # 1. Sum of vector
> vishal = c(1:10)
> sum(vishal)
[1] 55
>
> # 2. Mean
> mean(vishal)
[1] 5.5
>
> # 3. Product
> prod(vishal)
[1] 3628800
> ++
```

2) Write R Program to Take Input from User.



The screenshot shows the RStudio interface with a script editor on the left and the Environment, History, Connections, and Tutorial panels on the right. The script in the editor is as follows:

```
R 4.2.2 - ~/
> # 2. Take Input from user
> vishal2 = readline()
234
> vishal2
[1] "234"
> vishal2 = as.integer(vishal2)
> print(vishal2)
[1] 234
>
```

3) Write R Program to sample() function on given example.

A screenshot of the RStudio interface. The main editor window shows an R script with the following code:

```
> # 3. Sample Function in R
> vishal=c(23,45,21,34,5,6,7,8,86,45,3)
> print(sample(vishal,4))
[1] 86 8 7 23
> print(sample(vishal,1))
[1] 21
> print(sample(vishal,3))
[1] 23 7 21
>
```

The console on the right is empty. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The bottom status bar shows 'Project: (None)'.

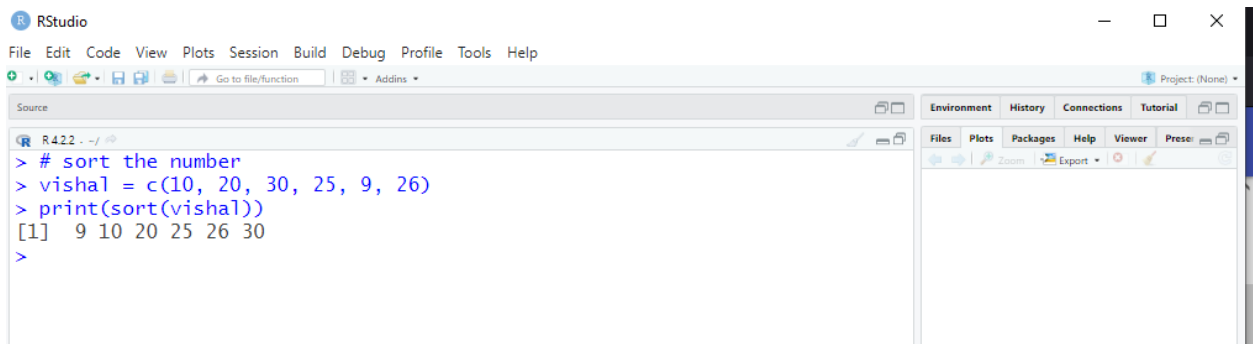
4) Write R Program to minimum, maximum and range of a given Vector.

A screenshot of the RStudio interface. The main editor window shows an R script with the following code:

```
> vishal = c(10, 20, 30, 40, 50, 60)
> print(vishal)
[1] 10 20 30 40 50 60
> print(paste(max(vishal)))
[1] "60"
> print(paste(min(vishal)))
[1] "10"
> |
```

The console on the right is empty. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The bottom status bar shows 'Project: (None)'.

5) Write R Program to Sort a given Vector.

A screenshot of the RStudio interface. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. Below the menu is a toolbar with icons for file operations and a search bar. The main editor window shows the following R code:

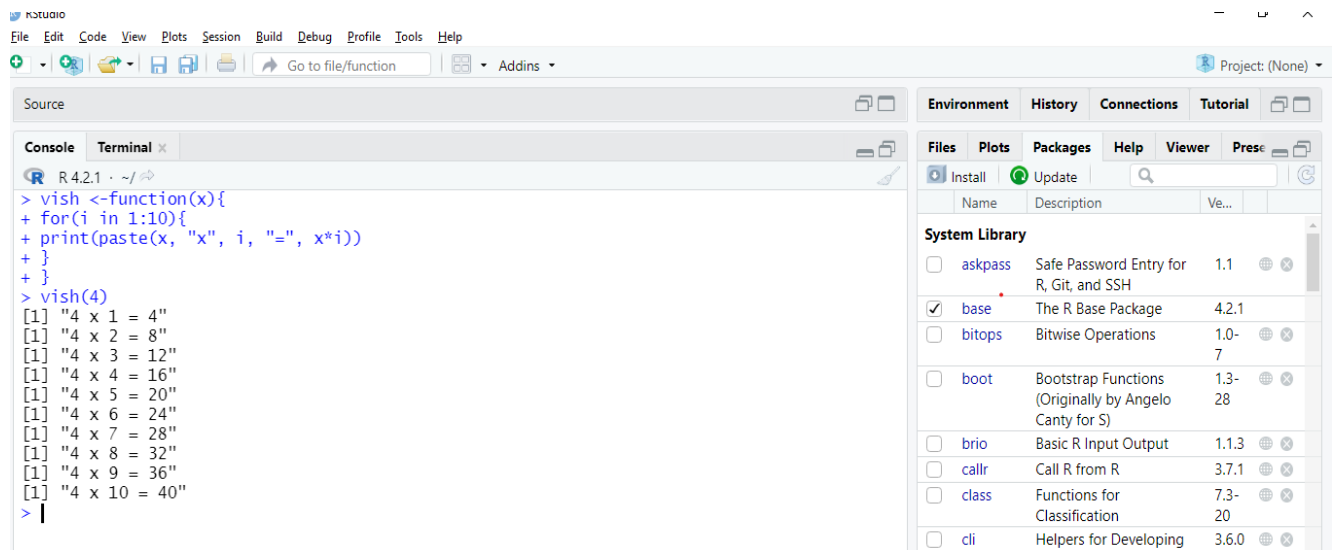
```
> # sort the number
> vishal = c(10, 20, 30, 25, 9, 26)
> print(sort(vishal))
[1] 9 10 20 25 26 30
>
```

The right-hand pane shows the Environment, History, Connections, and Tutorial tabs, all of which are currently empty.

6) Write R Program to Find the Factorial of a Number.

```
> factorial<-function(x){
+ f=1
+ if(x<0){
+ print("Sorry factorial does not exist")}
+ else if(x==0){
+ print("The factorial of 0 is 1")}
+ else{
+ for(i in 1:x){
+ f=f*i}
+ print(paste("The factorial of", x , "is",f))}
+ }
> factorial(0)
[1] "The factorial of 0 is 1"
> factorial(1)
[1] "The factorial of 1 is 1"
> factorial(-1)
[1] "Sorry factorial does not exist"
> factorial(2)
[1] "The factorial of 2 is 2"
> factorial(3)
[1] "The factorial of 3 is 6"
> factorial(4)
[1] "The factorial of 4 is 24"
> factorial(5)
[1] "The factorial of 5 is 120"
```

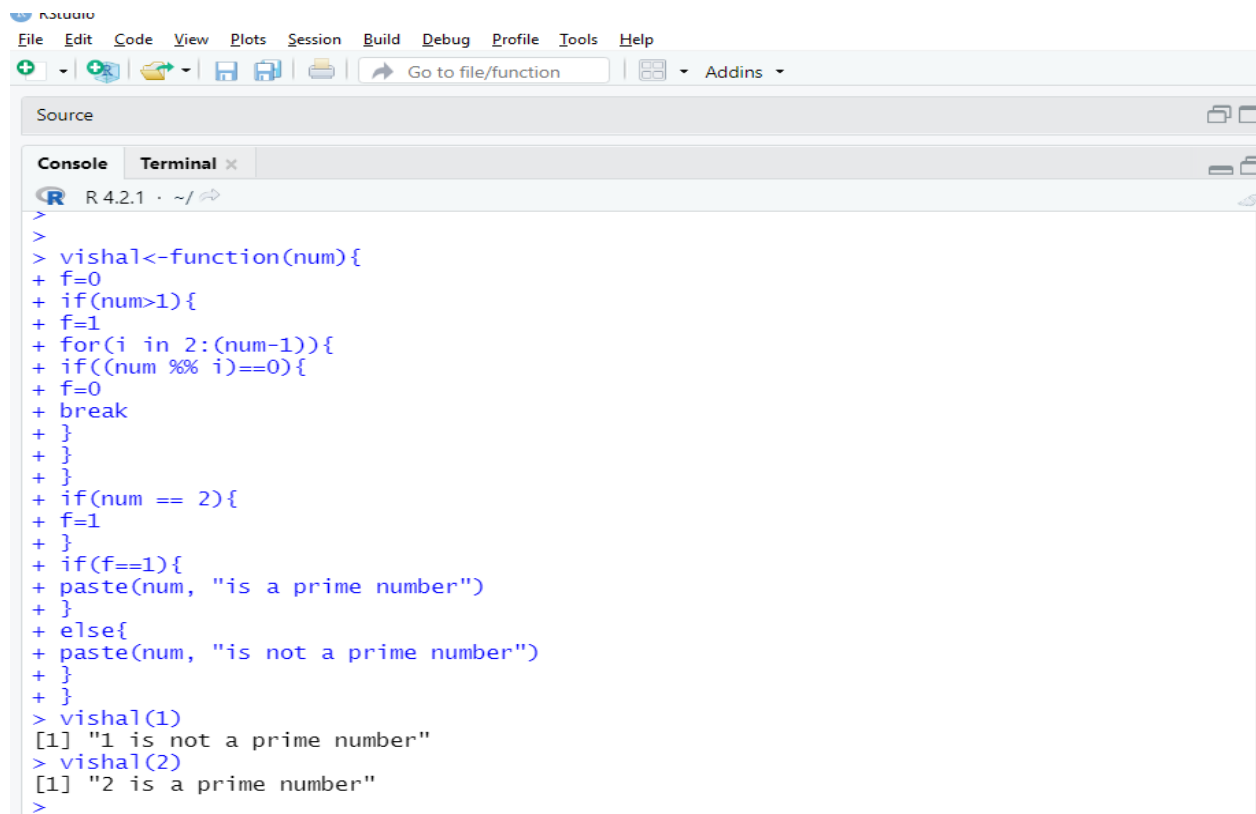
7) Write R Program to Print Table of given Number.



The screenshot shows the RStudio interface. The console window displays the execution of an R function named `vish`. The function takes a number `x` as input and prints a multiplication table for `x` from 1 to 10. The output for `vish(4)` is as follows:

```
> vish <-function(x){
+ for(i in 1:10){
+ print(paste(x, "x", i, "=", x*i))
+ }
+ }
> vish(4)
[1] "4 x 1 = 4"
[1] "4 x 2 = 8"
[1] "4 x 3 = 12"
[1] "4 x 4 = 16"
[1] "4 x 5 = 20"
[1] "4 x 6 = 24"
[1] "4 x 7 = 28"
[1] "4 x 8 = 32"
[1] "4 x 9 = 36"
[1] "4 x 10 = 40"
> |
```

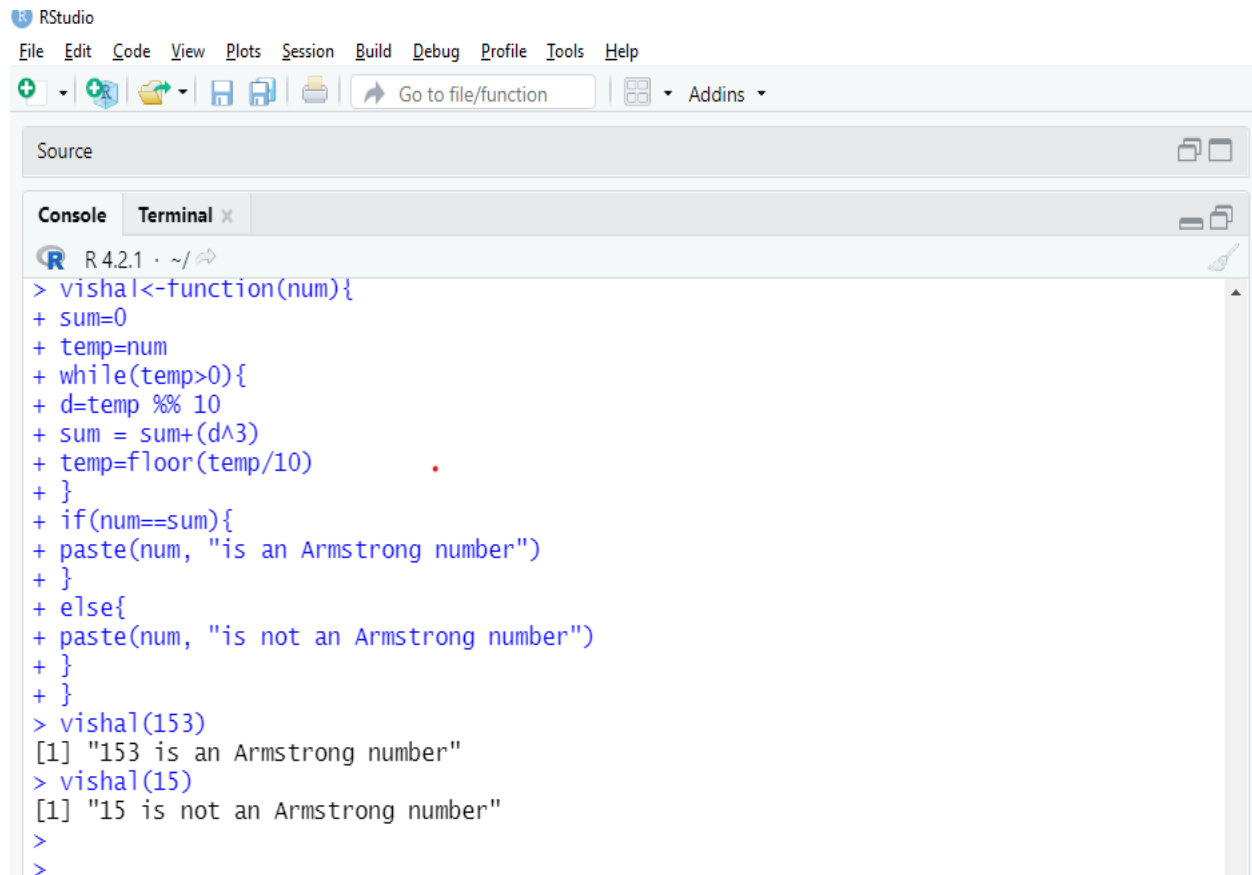
8) Write R Program to Check Prime Number.



The screenshot shows the RStudio interface. The console window displays the execution of an R function named `vishal`. The function takes a number `num` as input and checks if it is a prime number. The output for `vishal(1)` and `vishal(2)` is as follows:

```
> 
> 
> vishal<-function(num){
+ f=0
+ if(num>1){
+ f=1
+ for(i in 2:(num-1)){
+ if((num %% i)==0){
+ f=0
+ break
+ }
+ }
+ if(num == 2){
+ f=1
+ }
+ if(f==1){
+ paste(num, "is a prime number")
+ }
+ else{
+ paste(num, "is not a prime number")
+ }
+ }
> vishal(1)
[1] "1 is not a prime number"
> vishal(2)
[1] "2 is a prime number"
>
```

9) Write R Program to check Armstrong Number.



The screenshot shows the RStudio interface with the console pane active. The console displays the following R code and its output:

```
R 4.2.1 · ~/
> vishal<-function(num){
+ sum=0
+ temp=num
+ while(temp>0){
+ d=temp %% 10
+ sum = sum+(d^3)
+ temp=floor(temp/10)
+ }
+ if(num==sum){
+ paste(num, "is an Armstrong number")
+ }
+ else{
+ paste(num, "is not an Armstrong number")
+ }
+ }
> vishal(153)
[1] "153 is an Armstrong number"
> vishal(15)
[1] "15 is not an Armstrong number"
>
>
```

10) Write R Program to Print the Fibonacci Sequence.

```
Console Terminal x
R 4.2.1 · ~/
>
> vishal <- function(num){
+ a=0
+ b=1
+ c=2
+ if(num<=0){
+ print("Enter positive number")
+ }
+ else{
+ if(num==1){
+ print(a)
+ }
+ else{
+ print(a)
+ print(b)
+ while(c<num){
+ n=a+b
+ print(n)
+ a=b
+ b=n
+ c=c+1
+ }
+ }
+ }
+ }
> vishal(10)
[1] 0
[1] 1
[1] 1
[1] 2
[1] 3
[1] 5
[1] 8
[1] 13
[1] 21
[1] 34
> |
```

11) Write R Program to find nth highest value in a given vector.

```
Console Terminal x
R 4.2.1 · ~/
>
> vishal<-function(n){
+ vis<-c(15,36,24,67,86,2,34,5,6,33)
+ paste(n, " th highest value is ", sort(vis, TRUE)[n])
+ }
> vishal(4)
[1] "4 th highest value is 34"
> vishal(7)
[1] "7 th highest value is 15"
>
.
```

12) Write R Program to create a vector using : operator and seq() function.

```
Console Terminal x
R 4.2.1 · ~/
>
> vishal1<-c(1:10)
> vishal1
[1] 1 2 3 4 5 6 7 8 9 10
> vishal2<-c(seq(1,5, by=0.5))
> vishal2
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
>
>
>
>
```

13) Write R Program to convert a given list to vector.

```
Console Terminal x
R 4.2.1 · ~/
>
>
>
>
>
>
>
> vishal = list(1,2,3,4,5)
> vishal
[[1]]
[1] 1

[[2]]
[1] 2

[[3]]
[1] 3

[[4]]
[1] 4

[[5]]
[1] 5

> vishal1 = unlist(vishal)
> vishal1
[1] 1 2 3 4 5
> |
```

14) Write R Program for Convert Decimal into Binary using Recursion.

```
> d2b<-function(x){  
+ if(x>1){  
+ d2b(as.integer(x/2))}  
+ cat(x %% 2)}  
> d2b(8)  
1000  
> d2b(5)  
101
```

15) Write R Program to Find H.C.F. or G.C.D.

```
> hcf<-function(x,y){  
+ if(x>y){  
+ s=y}  
+ else{  
+ s=x}  
+ for(i in 1:s){  
+ if((x%%i==0) && (y%%i==0)){  
+ hcf=i}}  
+ return(hcf)}  
> hcf(4,3)  
[1] 1  
> hcf(72,120)  
[1] 24
```


16) Write 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.

```
> v1=c(1,2,3,4)
> v2=c(5,6,7,8,9,0)
> arr=array(c(v1,v2),dim=c(3,3,2))
> arr
, , 1
      [,1] [,2] [,3]
[1,]     1     4     7
[2,]     2     5     8
[3,]     3     6     9

, , 2
      [,1] [,2] [,3]
[1,]     0     3     6
[2,]     1     4     7
[3,]     2     5     8

> arr[2,,2]
[1] 1 4 7
> arr[3,3,1]
[1] 9
```

17) Write R Program to create two 2x3 matrix and add, subtract, multiply and divide the matrixes.

```

> m1=matrix(c(1,2,3,4,5,6),nrow=2)
> m2=matrix(c(7,8,9,10,11,12),nrow=2)
> m1+m2
      [,1] [,2] [,3]
[1,]    8   12   16
[2,]   10   14   18
> m1-m2
      [,1] [,2] [,3]
[1,]   -6   -6   -6
[2,]   -6   -6   -6
> m1*m2
      [,1] [,2] [,3]
[1,]    7   27   55
[2,]   16   40   72
> m1/m2
      [,1] [,2] [,3]
[1,] 0.1428571 0.3333333 0.4545455
[2,] 0.2500000 0.4000000 0.5000000

```

18) Write R Program to access the element at 3rd column and 2nd row, only the 3rd row and only the 4th column of a given matrix.

```

> m=matrix(c(1:16),nrow=4,byrow=TRUE)
> m
      [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
[4,]   13   14   15   16
> m[2,3]
[1] 7
> m[3,]
[1]  9 10 11 12
> m[,4]
[1]  4  8 12 16

```

19) Write R Program to find row and column index of maximum and minimum value in a given matrix.

```
> m=matrix(c(1:16),nrow=4)
> m
      [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16
> which(m==max(m),arr.ind=TRUE)
      row col
[1,]    4    4
> which(m==min(m),arr.ind=TRUE)
      row col
[1,]    1    1
```

20) Write R Program to concatenate two given matrices of same column but different rows.

```
> m1=matrix(1:12,ncol=3)
> m2=matrix(13:24,ncol=3)
> m1
      [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
> m2
      [,1] [,2] [,3]
[1,]   13   17   21
[2,]   14   18   22
[3,]   15   19   23
[4,]   16   20   24
> dim(rbind(m1,m2))
[1]  8  3
```

21) Write R Program to extract specific column from a data frame using column name.

```
> df=data.frame(
+ name=c("Abhay","Prateek","Vanshika","Neha","Nishi","Tushar","Balram","Sapna","Nikita","Gautam"),
+ score=c(95,70,85,80,83,72,60,67,75,80),
+ attempts=c(1,2,2,1,2,3,3,2,1,1),
+ qualify=c('y','n','y','y','y','n','n','n','n','n'))
> df
  name score attempts qualify
1  Abhay   95         1      y
2 Prateek   70         2      n
3 Vanshika  85         2      y
4   Neha   80         1      y
5   Nishi  83         2      y
6  Tushar  72         3      n
7  Balram  60         3      n
8   Sapna  67         2      n
9  Nikita  75         1      n
10 Gautam  80         1      n
> data.frame(df$name,df$score)
  df.name df.score
1  Abhay      95
2 Prateek      70
3 Vanshika     85
4   Neha     80
5   Nishi     83
6  Tushar     72
7  Balram     60
8   Sapna     67
9  Nikita     75
10 Gautam     80
```