**1.1) Calculate the cumulative sum("running total")of the numbers 2,3,4,5,6**

**Program :**

```
cumsum(2:6)
```

**Output:**

```
> cumsum(2:6)
[1]  2  5  9 14 20
```

**1.2) Print the 1 to 10 numbers in reverse order.**

**Program:**

```
rev(1:10)
```

**Output:**

```
> rev(1:10)
 [1] 10  9  8  7  6  5  4  3  2  1
```

**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.**

**Program :**

```
Name = readline(prompt = "Enter your name : ")
Age = readline(prompt = "Enter your age : ")
print(paste("My name is",Name,"and i am ",Age,"years old"))
print(R.version.string)
```

**Output:**

```
> Name = readline(prompt = "Enter your name : ")
Enter your name : Harry styles

> Age = readline(prompt = "Enter your age : ")
Enter your age : 25

> print(paste("My name is",Name,"and i am ",Age,"years old"))
[1] "My name is Harry styles and i am  25 years old"
> print(R.version.string)
[1] "R version 3.6.0 (2019-04-26)"
```

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

**Program:**

```
rm(list=ls())
Name = "R programming"
n1=1
n2=15.25
nums=c(10,5,21,65,12,80)
print(ls())
print("Details of the objects in memeory : ")
print(ls.str())
```

**Output**:

```
> rm(list=ls())
> Name = "R programming"
> n1=1
> n2=15.25
> nums=c(10,5,21,65,12,80)
> print(ls())
[1] "n1"   "n2"   "Name" "nums"
> print("Details of the objects in memeory : ")
[1] "Details of the objects in memeory : "
> print(ls.str())
n1 :  num 1
n2 :  num 15.2
Name :  chr "R programming"
nums :  num [1:6] 10 5 21 65 12 80
```

**2.3) 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.**

**Program :**

```
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:**

```
> print("Sequence of numbers from 20 to 50 : ")
[1] "Sequence of numbers from 20 to 50 : "
> print(seq(20,50))
 [1] 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37
[19] 38 39 40 41 42 43 44 45 46 47 48 49 50
> print("Mean of numbers from 20 to 60 : ")
[1] "Mean of numbers from 20 to 60 : "
> print(mean(20,60))
[1] 20
> print("Sum of numbers from 51 to 91 : ")
[1] "Sum of numbers from 51 to 91 : "
> print(sum(51,91))
[1] 142
```

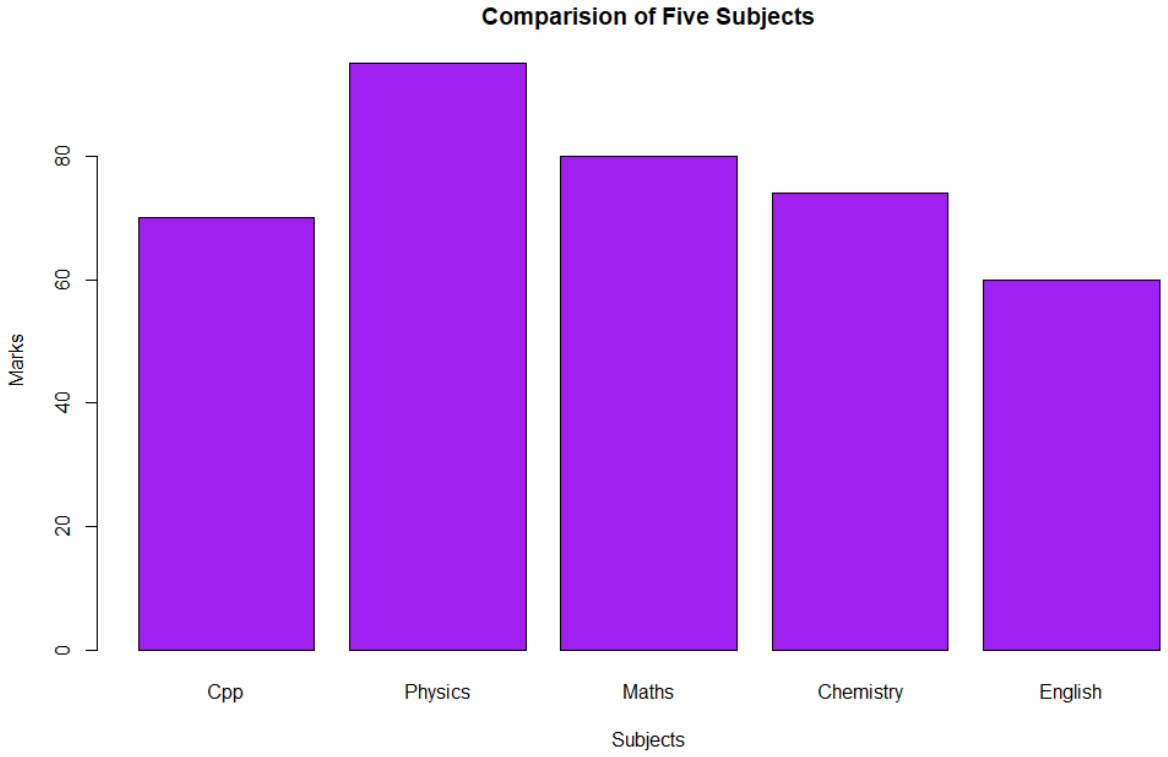
**3)Graphics**

**Write a R program to create a simple bar plot of five subjects marks.**

**Program:**

```
marks=c(70,95,80,74,60)
barplot(marks,main="Comparision of Five Subjects", xlab="Subjects" ,
ylab="Marks",names.arg=c("Cpp","Physics","Maths","Chemistry","English"),col="purple",h
oriz=FALSE)
```

**Output:**



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

**Program:**

```
str=c("Jin","Suga","RapMonster","Jhope","Jimin","V","Jungkook","Alekhya","Alekhya")
unique(str)
```

**Output:**

```
[1] "Jin"       "Suga"      "RapMonster" "Jhope"     "Jimin"
[6] "V"         "Jungkook"  "Alekhya"
```

**4.2) 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**

**Program:**

```
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 :"
> print(m)
     a b c
[1,] 1 4 7
[2,] 2 5 8
[3,] 3 6 9
```

**4.3) Write a R program to create a matrix from a list of given vectors.**

**Program:**

```
M=matrix((1:16) , nrow = 4 , byrow = TRUE)
print("Original Matrix")
print(M)
```

Output:

```
> M=matrix((1:16) , nrow = 4 , byrow = TRUE)

> print("Original Matrix")

[1] "Original Matrix"

> print(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
```

**5.1)Write a R program to append value to a given empty vector**

**Program :**

```
vector = c()
values = c(10,20,30,40,50,60,70,80,90,100)
for(i in 1:length(values))
        vector[i]=values[i]
print("Updated vector : ")
print(vector)
```

**Output:**

```
[1] "Updated vector : "

[1]  10  20  30  40  50  60  70  80  90 100
```

**5.2)Write a program to multiply 2 vectors of integers type and length 3**

**Program:**

```
x=c(10,20,30)
y=c(20,10,40)
print("Original Vectors : ")
print(x)
print(y)
print("Product of 2 Vectors :")
z = x*y
print(z)
```

Output:

```
> x=c(10,20,30)
> y=c(20,10,40)
> print("Original Vectors : ")
[1] "Original Vectors : "

> print(x)
[1] 10 20 30

> print(y)
[1] 20 10 40

> print("Product of 2 Vectors :")
[1] "Product of 2 Vectors :"
> z = x*y

> print(z)
[1]  200  200 1200
```

**5.3) Write a R program to find the Sum, Mean and Product of a Vector, ignore element like NA or NAN.**

**Program:**

```
x=c(10,NULL,20,30,NA)
print("Sum : ")
print(sum(x,na.rm = TRUE))
print("Mean : ")
print(mean(x,na.rm = TRUE))
print("Product : ")
print(prod(x,na.rm = TRUE))
```

**Output:**

```
> x=c(10,NULL,20,30,NA)
> print("Sum : ")
[1] "Sum : "

> print(sum(x,na.rm = TRUE))
[1] 60

> print("Mean : ")
[1] "Mean : "

> print(mean(x,na.rm = TRUE))
[1] 20

> print("Product : ")
[1] "Product : "

> print(prod(x,na.rm = TRUE))
[1] 6000
```

**6.1) 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 x 2 matrix with labels and fill the matrix by columns.**

**Program:**

```
m1 = matrix(1:20 , nrow = 5,  ncol = 4)
print("5 x 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 x 3 matrix with labels filled by rows : ")
print(m2)
cells = c(9,11,12,14)
rnames = c("Row1","Row2")
cnames = c("Col1","Col2")
m3 = matrix(cells,nrow = 2, ncol=2 ,byrow=TRUE, dimnames = list(rnames,cnames))
print("2 x 2 matrix with labels filled by rows : ")
print(m3)
```

**Output:**

```
> m1 = matrix(1:20 , nrow = 5, ncol = 4)
> print("5 x 4 Matrix : ")
[1] "5 x 4 Matrix : "
> print(m1)
     [,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
> 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 x 3 matric with labels filled by rows : ")
[1] "3 x 3 matric with labels filled by rows : "
> print(m2)
     Col1 Col2 Col3
Row1    1    3    5
Row2    7    8    9
Row3   11   12   14
> rnames = c("Row1","Row2")
> cnames = c("Col1","Col2")
> m3 = matrix(cells,nrow = 2, ncol=2 ,byrow=TRUE, dimnames = list(rnames,cnames))
> print("2 x 2 matric with labels filled by rows : ")
[1] "2 x 2 matric with labels filled by rows : "
> print(m3)
     Col1 Col2
Row1    9   11
Row2   12   14
```

**6.2) Write a R program to create a two-dimensional 5x3 array of sequence of even integers greater than 50.**

**Program:**

```
a = array(seq(from = 50, length.out=15, by=2), c(5,3))
print("Content of the array : ")
print("5 x 3 array of sequence of the even integers greater than 50 : ")
print(a)
```

**Output:**

```
> a = array(seq(from = 50, length.out=15, by=2), c(5,3))
> print("Content of the array : ")
[1] "Content of the array : "

> print("5 x 3 array of sequence of the even integers greater than 50 : ")
[1] "5 x 3 array of sequence of the even integers greater than 50 : "

> print(a)
     [,1] [,2] [,3]
[1,]   50   60   70
[2,]   52   62   72
[3,]   54   64   74
[4,]   56   66   76
[5,]   58   68   78
```

**6.3) Write a R program to find row and column index of maximum and minimum value in a given matrix**

**Program:**

```
m = matrix(c(1:16),nrow = 4,byrow = TRUE)
print("Original Matrix: ")
print(m)
result = which(m == max(m),arr.ind=TRUE)
print("Row of maximum value of the said matrix: ")
print(result)
result = which(m == min(m),arr.ind=TRUE)
print("Row of minimum value of the said matrix: ")
print(result)
```

**Output:**

```
> m = matrix(c(1:16),nrow = 4,byrow = TRUE)
> print("Original Matrix: ")
[1] "Original Matrix: "
> print(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
> result = which(m == max(m),arr.ind=TRUE)
> print("Row of maximum value of the said matrix: ")
[1] "Row of maximum value of the said matrix: "
> print(result)
     row col
[1,]   4   4
> result = which(m == min(m),arr.ind=TRUE)
> print("Row of minimum value of the said matrix: ")
[1] "Row of minimum value of the said matrix: "
> print(result)
     row col
[1,]   1   1
```

**7)Arrays**

**7.1) Write a R program to combine three arrays so that the first row of the first array is followed by the first row of the second array and then first row of the third array.**

**Program:**

```
num1=rbind(rep("A",3),rep("B",3),rep("C",3))
print("num1")
print(num1)
num2=rbind(rep("P",3),rep("Q",3),rep("R",3))
print("num2")
print(num2)
num3=rbind(rep("X",3),rep("Y",3),rep("Z",3))
print("num3")
print(num3)
a = matrix(t(cbind(num1,num2,num3)),ncol=3,byrow=T)
print("Combine three arrays,taking one row for each one by one: ")
print(a)
```

**Output:**

```
[1] "num1"
     [,1] [,2] [,3]
[1,] "A"  "A"  "A"
[2,] "B"  "B"  "B"
[3,] "C"  "C"  "C"
[1] "num2"
     [,1] [,2] [,3]
[1,] "P"  "P"  "P"
[2,] "Q"  "Q"  "Q"
[3,] "R"  "R"  "R"
[1] "num3"
     [,1] [,2] [,3]
[1,] "X"  "X"  "X"
[2,] "Y"  "Y"  "Y"
[3,] "Z"  "Z"  "Z"
[1] "Combine three arrays,taking one row for each one by one: "
      [,1] [,2] [,3]
 [1,] "A"  "A"  "A"
 [2,] "P"  "P"  "P"
 [3,] "X"  "X"  "X"
 [4,] "B"  "B"  "B"
 [5,] "Q"  "Q"  "Q"
 [6,] "Y"  "Y"  "Y"
 [7,] "C"  "C"  "C"
 [8,] "R"  "R"  "R"
 [9,] "Z"  "Z"  "Z"
```

**7.2)Write a R program to create an array using four given columns, three given rows, and two given tables and display the content of the array.**

**Program:**

```
array1 = array(1:30, dim=c(3,5,2))
print(array1)
```

**Output:**

```
> array1 = array(1:30, dim=c(3,5,2))
> print(array1)
, , 1

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    4    7   10   13
[2,]    2    5    8   11   14
[3,]    3    6    9   12   15

, , 2

     [,1] [,2] [,3] [,4] [,5]
[1,]   16   19   22   25   28
[2,]   17   20   23   26   29
[3,]   18   21   24   27   30
```

**8)Data frame-I**

**8.1) Write a R program to create an empty data frame.**

**Program:**

```
df=data.frame(Ints=integer(),Doubles=double(),characters=character(),
 Logicals=logical(),Factors=factor(),stringsAsFactors = FALSE)
print("Structures of the empty dataframe: ")
print(str(df))
```

**OUTPUT:**

```
[1] "Structures of the empty dataframe: "
'data.frame': 0 obs. of 5 variables:
$ Ints : int
$ Doubles : num
$ characters: chr
$ Logicals : logi
$ Factors : Factor w/ 0 levels:
NULL
```

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

```
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)
```

**OUTPUT:**

```
[1] "Original data frame: "
[1] "Anastasia" "Dima" "Katherine" "James" "Emily" "Michael"
"Matthew" "Laura"
[9] "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
```

**9) Data frame-II**

**9.1) Write a R program to create a data frame using two given vectors and display the duplicated elements and unique rows of the said data frame.**

**Program :**

```
a=c(10,20,40,10,20,40)
b=c(10,20,30,10,20,40)
print("original data frame")
df=data.frame(a,b)
print(df)
print(" Data frame with duplicated values")
print(duplicated(df))
print("Data frame with unique values")
print(unique(df))
```

**Output :**

```
[1] "original data frame"
 a b
1 10 10
2 20 20
3 40 30
4 10 10
5 20 20
6 40 40
[1] " Data frame with duplicated values"
[1] FALSE FALSE FALSE TRUE TRUE FALSE
[1] "Data frame with unique values"
 a b
1 10 10
2 20 20
3 40 30
6 40 40
```

**9.2) Write a R program to save the information of a data frame in a file and display the information of the file.**

**Program :**

```
exam_data=data.frame(
name=c('Anastasia','Dima','Katherine','James','Emily','Michael','Matthew','Laura','Kevin','Jonas'),
score=c(12.5,9,12,20,5,28,15,13,26.5,28),
attempts=c(1,3,2,3,1,1,2,3,2,1),
qualify=c('yes','no','yes','yes','yes','no','no','no','yes','yes'))
print("Original dataframe : ")
print(exam_data)
print("saving the information of the data frame in a file")
save(exam_data,file="data.cse")
load("data.cse")
print("Display the information of")
file.info("data.cse")
```

**Output:**

```
name Score attempts qualify
1 Anastasia 12.5 1 yes
2 Dima 9.0 3 no
3 Katherine 12.0 2 yes
4 James 20.0 3 yes
5 Emily 5.0 1 yes
6 Michael 28.0 1 no
7 Matthew 15.0 2 no
8 Laura 13.0 3 no
9 Kevin 26.5 2 yes
10 Jonas 28.0 1 yes
[1] "saving the information of the data frame in a file"
[1] "Display the information of"
 size isdir mode mtime ctime atime
data.cse 374 FALSE 666 2022-05-12 12:36:56 2022-05-10 13:54:05 2022-05-12 12:36:56
 exe
data.cse no
```

**10)Lists**

**10.1) Write a R program to create a list containing a vector, a matrix and a list and give names to the elements in the list. Access the first and second element of the list.**

**Program:**

```
list_data<-list(c("Red","Green","Black"),matrix(c(1,3,5,7,9,11),nrow
2),list("Python","PHP","Java"))
print("List:")
print(list_data)
names(list_data)=c("Color","Odd","Language(s)")
print("List with coloumn names:")
print(list_data)
print("1st element:")
print(list_data[1])
print("2nd element:")
print(list_data[2])
```

**Output:**

```
> list_data<-list(c("Red","Green","Black"),matrix(c(1,3,5,7,9,11),nrow =
2),list("Python","PHP","Java"))
> print("List:")
[1] "List:"
> print(list_data)
[[1]]
[1] "Red"   "Green" "Black"
[[2]]
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    3    7   11
[[3]]
[[3]][[1]]
[1] "Python"
[[3]][[2]]
[1] "PHP"
[[3]][[3]]
[1] "Java"
> names(list_data)=c("Color","Odd","Language(s)")
> print("List with coloumn names:")
[1] "List with coloumn names:"
> print(list_data)
$Color
[1] "Red"   "Green" "Black"
$Odd
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    3    7   11
$`Language(s)`
$`Language(s)`[[1]]
[1] "Python"
```

```
$`Language(s)`[[2]]
[1] "PHP"
$`Language(s)`[[3]]
[1] "Java"
> print("1st element:")
[1] "1st element:"
> print(list_data[1])
$Color
[1] "Red"   "Green" "Black"
> print("2nd element:")
[1] "2nd element:"
> print(list_data[2])
$Odd
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    3    7   11
```

**10.2) Write a R program to create a list containing a vector, a matrix and a list and remove the second element.**

**Program:**

```
list_data<-list(c("Red","Green","Black"),matrix(c(1,3,5,7,9,11),nrow =
2),list("Python","PHP","Java"))
print("List:")
print(list_data)
print("Remove the second element of the list:")
list_data[2]=NULL
print("New list:")
print(list_data)
```

**Output:**

```
> list_data<-list(c("Red","Green","Black"),matrix(c(1,3,5,7,9,11),nrow =
2),list("Python","PHP","Java"))
> print("List:")
[1] "List:"
> print(list_data)
[[1]]
[1] "Red"   "Green" "Black"
[[2]]
     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    3    7   11
[[3]]
[[3]][[1]]
[1] "Python"
[[3]][[2]]
[1] "PHP"
[[3]][[3]]
[1] "Java"
> print("Remove the second element of the list:")
[1] "Remove the second element of the list:"
> list_data[2]=NULL
> print("New list:")
[1] "New list:"
> print(list_data)
[[1]]
[1] "Red"   "Green" "Black"
[[2]]
[[2]][[1]]
[1] "Python"
[[2]][[2]]
[1] "PHP"
[[2]][[3]]
[1] "Java"
```

**10.3) Write a R program to select second element of a given nested list.**

**Program:**

```
x=list(list(0,2),list(3,4),list(5,6))
print("Original nested list:")
print(x)
e=lapply(x,'[[',2)
print("second element of the nested list")
print(e)
```

**Output:**

```
> x=list(list(0,2),list(3,4),list(5,6))
> print("Original nested list:")
[1] "Original nested list:"
> print(x)
[[1]]
[[1]][[1]]
[1] 0
[[1]][[2]]
[1] 2
[[2]]
[[2]][[1]]
[1] 3
[[2]][[2]]
[1] 4
[[3]]
[[3]][[1]]
[1] 5
[[3]][[2]]
[1] 6
> e=lapply(x,'[[',2)
> print("second element of the nested list")
[1] "second element of the nested list"
> print(e)
[[1]]
[1] 2
[[2]]
[1] 4
[[3]]
[1] 6
```

**11)Lists-continued**

**11.1) Write a R program to merge two given lists into one list**

**Program:**

```
n1=list(1,2,3)
c1=list("Red","Green","Black")
print("Original lists:")
print(n1)
print(c1)
print("Merge the said lists:")
mlist=c(n1,c1)
print("New Merged list:")
print(mlist)
```

**Output:**

```
> n1=list(1,2,3)
> c1=list("Red","Green","Black")
> print("Original lists:")
[1] "Original lists:"
> print(n1)
[[1]]
[1] 1
[[2]]
[1] 2
[[3]]
[1] 3
> print(c1)
[[1]]
[1] "Red"
[[2]]
[1] "Green"
[[3]]
[1] "Black"
> print("Merge the said lists:")
[1] "Merge the said lists:"
> mlist=c(n1,c1)
> print("New Merged list:")
[1] "New Merged list:"
[[1]]
[1] 1
[[2]]
[1] 2
[[3]]
[1] 3
[[4]]
[1] "Red"
[[5]]
[1] "Green"
[[6]][1] "Black"
```

**11.2) Write a R program to create a list named s containing sequence of 15 capital letters, starting from ‘E’.**

**Program:**

```
s=LETTERS[match("E",LETTERS):(match("E",LETTERS)+15)]
print("Sequence of 15 captial letters, starting from 'E'-")
print(s)
```

**Output:**

```
> s=LETTERS[match("E",LETTERS):(match("E",LETTERS)+15)]
> print("Sequence of 15 captial letters, starting from 'E'-")
[1] "Sequence of 15 captial letters, starting from 'E'-"
> print(s)
 [1] "E" "F" "G" "H" "I" "J" "K" "L" "M" "N" "O" "P" "Q" "R" "S" "T"
```

**11.3) Write a R program to assign new names "a", "b" and "c" to the elements of a given list.**

**Program:**

```
list1=list(g1=1:10,g2="R Programming",g3="HTML")
print("Original list:")
print(list1)
names(list1)=c("a","b","c")
print("Assign new names 'a','b' and 'c'to the elements in said list")
print(list1)
```

**Output:**

```
> list1=list(g1=1:10,g2="R Programming",g3="HTML")
> print("Original list:")
[1] "Original list:"
> print(list1)
$g1
 [1]  1  2  3  4  5  6  7  8  9 10
$g2
[1] "R Programming"
$g3
[1] "HTML"
> names(list1)=c("a","b","c")
> print("Assign new names 'a','b' and 'c'to the elements in said list")
[1] "Assign new names 'a','b' and 'c'to the elements in said list"
> print(list1)
$a
 [1]  1  2  3  4  5  6  7  8  9 10
$b
[1] "R Programming"
$c
[1] "HTML"
```

**12)Factors**

**12.1) Write a R program to find the levels of factor of a given vector**

**Program:**

```
v=c(1,2,3,4,NA,3,2,4,5,NA,5)
print("Original vector:")
print(v)
print("Levels of factor of the said vector.")
print(levels(factor(v)))
```

**Output:**

```
> v=c(1,2,3,4,NA,3,2,4,5,NA,5)
> print("Original vector:")
[1] "Original vector:"
> print(v)
 [1]  1  2  3  4 NA  3  2  4  5 NA  5
> print("Levels of factor of the said vector.")
[1] "Levels of factor of the said vector."
> print(levels(factor(v)))
[1] "1" "2" "3" "4" "5"
```

**12.2) Write a R program to create an ordered factor from data consisting of the names of months.**

**Program:**

```
mons_v=c("March","April","January","November","January","September","October","Septe
mber","November","August","February","January","November","November","February","M
ay","August","February","July","December","August","August","September","November","
September","February","April")
print("Original vector:")
print(mons_v)
f=factor(mons_v)
print("Ordered factors of the said vector:")
print(f)
print(table(f))
```

**Output:**

```
>mons_v=c("March","April","January","November","January","September","October","Sept
ember","November","August","February","January","November","November","February","
May","August","February","July","December","August","August","September","November",
"September","February","April")
> print("Original vector:")
[1] "Original vector:"
> print(mons_v)
 [1] "March"     "April"     "January"   "November"  "January"   "September" "October"
 [8] "September" "November"  "August"    "February"  "January"   "November"  "November"
[15] "February"  "May"       "August"    "February"  "July"      "December"  "August"
[22] "August"    "September" "November"  "September" "February"  "April"
> f=factor(mons_v)
> print("Ordered factors of the said vector:")
[1] "Ordered factors of the said vector:"
> print(f)
 [1] March     April     January   November  January   September October   September
November
[10] August    February  January   November  November  February  May       August
February
[19] July      December  August    August    September November  September February  April
Levels: April August December February January July March May November October
September
> print(table(f))
f
    April    August  December  February   January      July     March       May  November
        2         4         1         4         3         1         1         1         5
  October September
        1         4
```

**12.3) Write a R program to concatenate two given factor in a single factor.**

**Program:**

```
f1<-factor(sample(LETTERS,size = 6,replace = TRUE))
f2<-factor(sample(LETTERS,size = 6,replace = TRUE))
print("Original factors:")
print(f1)
print(f2)
f=factor(c(levels(f1)[f1],levels(f2)[f2]))
print(f)
```

OUTPUT:

```
> f1<-factor(sample(LETTERS,size = 6,replace = TRUE))
> f2<-factor(sample(LETTERS,size = 6,replace = TRUE))
> print("Original factors:")
[1] "Original factors:"
> print(f1)
[1] F S M E M L
Levels: E F L M S
> print(f2)
[1] L F V C R O
Levels: C F L O R V
> f=factor(c(levels(f1)[f1],levels(f2)[f2]))
> print(f)
[1] F S M E M L L F V C R O
Levels: C E F L M O R S V
```

**14)The weights of five people are given before and after a diet programme are given in the**
**table.**
**Before 72 72 78 79 105**
**After 67 65 79 70 93**
**Read the Before and after values into two different vectors called before and after .use R to**
**evaluate**
**the amount of weight lost for each participant. what is the average amount of weight lost?**

**PROGRAM:**

```
print("Before diet programme")
before<-c(78,72,78,79,105)
print("After diet programme")
after<-c(67,65,79,70,93)
print("Weight lost")
d<-before-after
print(d)
sum(d)
print("Average Weight lost")
e<-sum(d)/5
print(e)
```

**output:**

```
> print("Before diet programme")
[1] "Before diet programme"
> before<-c(78,72,78,79,105)
> print("After diet programme")
[1] "After diet programme"
> after<-c(67,65,79,70,93)
> print("Weight lost")
[1] "Weight lost"
> d<-before-after
> print(d)
[1] 11 7 -1 9 12
> sum(d)
[1] 38
> print("Average Weight lost")
[1] "Average Weight lost"
> e<-sum(d)/5
> print(e)
[1] 7.6
```

**15)consider A=matrix(c(2,0,1,3),ncol-2)and B=matrix(c(5,2,4,-1),ncol=2)**
**a.find A+B**
**b.find A-B**

**Program:**

```
A<-matrix(c(2,0,1,3),ncol=2)
B<-matrix(c(5,2,4,-1),ncol=2)
A+B
A-B
```

**Output:**

```
> A<-matrix(c(2,0,1,3),ncol=2)
> B<-matrix(c(5,2,4,-1),ncol=2)
> A+B
 [,1] [,2]
[1,] 7 5
[2,] 2 2
> A-B
 [,1] [,2]
[1,] -3 -3
[2,] -2 4
```