Assignment No : 1
Assignment Name : To perform the basic mathematical operations in R-programming.
Name :
Roll No :

- **Assignment operator :-**

1. **leftward assignment**

```
x<-20
print(x)
x<<-30
print(x)
```

2. **rightward assignment**

```
40->y
print(y)
50->>y
print(y)
```

3. **Isqualto**

```
x=100
print(x)
```

OUTPUT:

```
[1] 20
[1] 30
[1] 40
[1] 50
[1] 100
```

- **Basic Arithmetic :-**

```
10+9/5^2
10+9/(5^2)
10+(9/5)^2
(10+9)/5^2
```

OUTPUT:

```
[1] 10.4
[1] 10.4
[1] 13.2
[1] 0.76
```

#Option

```
1/7
options(digits=3)
```

1/7

```
[1] 0.1428571
[1] 0.143
```

#pi

Pi

options(digits=22)

pi

```
[1] 3.14
[1] 3.141592653589793115998
```

#Regular Division

54/4

```
[1] 13.5
```

#integer division

54%/%4

```
[1] 13
```

#modulo(remainder)

54%%4

```
[1] 2
```

➢ Miscellaneous Mathematical function

x<-20

abs(x) #absolute value

sqrt(x) #square root

exp(x) #exponential transformation

log(x) #logarithmic transformation

cos(x) #trigonometric functions

OUTPUT:

```
[1] 20
[1] 4.47
[1] 4.85e+08
[1] 3
[1] 0.408
```

➢ infine and NaN Numbers :-

1/0 #infinity

```
[1]Inf
```

Inf-Inf #infinity minus infinity

```
[1] NaN
```

```
x<-2
y<-3
ls()
```

```
[1] "x" "y"
```

```
exists("x")  # identify
```

```
[1] TRUE
```

```
rm(x)  # remove defined object
rm(x,y)    # remove multiple objects
rm(list=ls())  #basically removes everything
```

Assignment No : 2.1
Assignment Name : Write program for Creating and Manipulating R Objects in R – Vectors.
Name :
Roll No :

❖ Creation of R-Vector :

#Single Element Vector

#Atomic vector of type character.

```
print("abc");
```

 # Atomic vector of type double.

```
print(12.5)
```

 # Atomic vector of type integer.

```
print(63L)
```

 # Atomic vector of type logical.

```
print(TRUE)
```

 # Atomic vector of type complex.

```
print(2+3i)
```

 # Atomic vector of type raw.

```
print(charToRaw('hello'))
```

OUTPUT:

```
[1] "abc"
[1] 12.5
[1] 63
[1] TRUE
[1] 2+3i
[1] 68 65 6c 6c 6f
```

#Multiple Elements Vector

```
v <- 5:13
print(v)
```

```
[1]  5  6  7  8  9 10 11 12 13
```

#Using sequence (Seq.) operator

```
print(seq(5, 9, by=0.4))
```

```
[1] 5.0 5.4 5.8 6.2 6.6 7.0 7.4 7.8 8.2 8.6 9.0
```

#Using the c() function

```
s <- c('apple','red',5,TRUE)
print(s)
```

```
[1] "apple" "red"   "5"     "TRUE"
```

❖ Manipulation of R-Vectors :

 # Create two vectors.

```
v1 <- c(3,8,4,5,0,11)
v2 <- c(4,11,0,8,1,2)
```

 # Vector addition.

```
add.result <- v1+v2
print(add.result)
```

 # Vector substraction.

```
sub.result <- v1-v2
print(sub.result)
```

 # Vector multiplication.

```
multi.result <- v1*v2
print(multi.result)
```

 # Vector division.

```
divi.result <- v1/v2
print(divi.result)
```

OUTPUT:

```
[1]  7 19  4 13  1 13
[1] -1 -3  4 -3 -1  9
[1] 12 88  0 40  0 22
[1] 0.750 0.727   Inf 0.625 0.000 5.500
```

Assignment No : 2.2
Assignment Name : Write program for Creating and Manipulating R Objects in R –Matrices.
Name :
Roll No :

❖ Creating R-Matrices :

 # Elements are arranged sequentially by row.

```
M <- matrix(c(3:14), nrow = 4, byrow = TRUE)

print(M)
```

OUTPUT:

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

 # Elements are arranged sequentially by column.

```
N <- matrix(c(3:14), nrow = 4, byrow = FALSE)

print(N)
```

OUTPUT:

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

 # Define the column and row names.

```
rownames = c("row1", "row2", "row3", "row4")

colnames = c("col1", "col2", "col3")

P <- matrix(c(3:14), nrow = 4, byrow = TRUE, dimnames = list(rownames, colnames))

print(P)
```

OUTPUT:

```
      col1 col2 col3
row1    3    4    5
row2    6    7    8
row3    9   10   11
row4   12   13   14
```

❖ Manipulating R-Matrices :

#Matrix Addition & Subtraction

 # Create two 2x3 matrices.

```
matrix1 <- matrix(c(3, 9, -1, 4, 2, 6), nrow = 2)

print(matrix1)
```

OUTPUT:

```
     [,1] [,2] [,3]
[1,]    3   -1    2
[2,]    9    4    6
```

```
matrix2 <- matrix(c(5, 2, 0, 9, 3, 4), nrow = 2)

print(matrix2)
```

OUTPUT:

```
     [,1] [,2] [,3]
[1,]    5    0    3
[2,]    2    9    4
```

 # Add the matrices.

```
result <- matrix1 + matrix2

cat("Result of addition","\n")

print(result)
```

OUTPUT:

```
     [,1] [,2] [,3]
[1,]    8   -1    5
[2,]   11   13   10
```

 # Subtract the matrices

```
result <- matrix1 - matrix2

cat("Result of subtraction","\n")

print(result)
```

OUTPUT:

```
     [,1] [,2] [,3]
[1,]   -2   -1   -1
[2,]    7   -5    2
```

 # Multiply the matrices.

```
result <- matrix1 * matrix2

cat("Result of multiplication","\n")
```

```
print(result)
```

OUTPUT:

```
     [,1] [,2] [,3]
[1,]   15    0    6
[2,]   18   36   24
```

 # Divide the matrices

```
result <- matrix1 / matrix2

cat("Result of division","\n")

print(result)
```

OUTPUT:

```
     [,1]  [,2]  [,3]
[1,]  0.6  -Inf 0.667
[2,]  4.5 0.444 1.500
```

Assignment No : 2.3
Assignment Name : Write program for Creating and Manipulating R Objects in R – Arrays.
Name :
Roll No :

❖ Creation of Array :

 # Create two vectors of different lengths.

```
vector1 <- c(5,9,3)

vector2 <- c(10,11,12,13,14,15)
```

 # Take these vectors as input to the array.

```
result <- array(c(vector1,vector2),dim = c(3,3,2))

print(result)
```

OUTPUT:

```
, , 1

     [,1] [,2] [,3]
[1,]    5   10   13
[2,]    9   11   14
[3,]    3   12   15

, , 2

     [,1] [,2] [,3]
[1,]    5   10   13
[2,]    9   11   14
[3,]    3   12   15
```

❖ Manipulating Array Elements :

 # Create two vectors of different lengths.

```
vector3 <- c(9,1,0)

vector4 <- c(6,0,11,3,14,1,2,6,9)

array2 <- array(c(vector1,vector2),dim = c(3,3,2))
```

 # create matrices from these arrays.

```
matrix1 <- array1[,,2]

matrix2 <- array2[,,2]
```

 # Add the matrices.

```
result <- matrix1+matrix2

print(result)
```

OUTPUT:

```
     [,1] [,2] [,3]
[1,]   10   20   26
[2,]   18   22   28
[3,]    6   24   30
```

Assignment No : 2.4
Assignment Name : Write program for Creating and Manipulating R Objects in R – DataFrames.
Name :
Roll No :

#Creating R -object in R-DataFrames

```
emp.data <- data.frame(
  emp_id = c (1:5),
  emp_name = c("Rick","Dan","Michelle","Ryan","Gary"),
  salary = c(623.3,515.2,611.0,729.0,843.25),

  start_date = as.Date(c("2012-01-01", "2013-09-23", "2014-11-15", "2014-05-11",
                         "2015-03-27")),
  stringsAsFactors = FALSE
)
```

 # Print the data frame.

```
print(emp.data)
```

OUTPUT:

```
  emp_id emp_name salary start_date
1      1     Rick    623 2012-01-01
2      2      Dan    515 2013-09-23
3      3 Michelle    611 2014-11-15
4      4     Ryan    729 2014-05-11
5      5     Gary    843 2015-03-27
```

 # Get the structure of the data frame.

```
str(emp.data)
```

OUTPUT:

```
'data.frame':     5 obs. of  4 variables:
 $ emp_id    : int  1 2 3 4 5
 $ emp_name  : chr  "Rick" "Dan" "Michelle" "Ryan" ...
 $ salary    : num  623 515 611 729 843
 $ start_date: Date, format: "2012-01-01" "2013-09-23" ...
```

 # Print the summary.

```
print(summary(emp.data))
```

OUTPUT:

```
   emp_id    emp_name          salary     start_date
 Min.   :1   Length:5         Min.   :515   Min.   :2012-01-01
 1st Qu.:2   Class :character   1st Qu.:611   1st Qu.:2013-09-23
 Median :3   Mode  :character   Median :623   Median :2014-05-11
 Mean   :3                      Mean   :664   Mean   :2014-01-14
 3rd Qu.:4                      3rd Qu.:729   3rd Qu.:2014-11-15
 Max.   :5                      Max.   :843   Max.   :2015-03-27
```

#Extract Data from Data Frame

 # Extract Specific columns.

```
result <- data.frame(emp.data$emp_name,emp.data$salary)

print(result)
```

OUTPUT:

```
  emp.data.emp_name emp.data.salary
1             Rick            623
2              Dan            515
3          Michelle            611
4             Ryan            729
5             Gary            843
```

 # Extract first two rows.

```
result <- emp.data[1:2,]

print(result)
```

OUTPUT:

```
  emp_id emp_name salary start_date
1      1     Rick    623 2012-01-01
2      2      Dan    515 2013-09-23
```

 # Extract 3rd and 5th row with 2nd and 4th column.

```
result <- emp.data[c(3,5),c(2,4)]

print(result)
```

OUTPUT:

```
emp_name start_date
3 Michelle 2014-11-15
5     Gary 2015-03-27
```

#Expand Data Frame

 # Add the coulmn.

```
emp.data$dept <- c("IT","Operations","IT","HR","Finance")

v <- emp.data

print(v)
```

OUTPUT:

```
emp_id emp_name salary start_date      dept
1      1    Rick    623 2012-01-01        IT
2      2     Dan    515 2013-09-23 Operations
3      3 Michelle    611 2014-11-15        IT
4      4    Ryan    729 2014-05-11        HR
5      5    Gary    843 2015-03-27    Finance
```

 # Create the second data frame

```
emp.newdata <-            data.frame(

emp_id = c (6:8),

emp_name = c("Rasmi","Pranab","Tusar"),

salary = c(578.0,722.5,632.8),

start_date = as.Date(c("2013-05-21","2013-07-30","2014-06-17")),

dept = c("IT","Operations","Fianance"),

stringsAsFactors = FALSE)
```

 # Bind the two data frames.

```
emp.finaldata <- rbind(emp.data,emp.newdata)

print(emp.finaldata)
```

OUTPUT:

```
  emp_id emp_name salary start_date      dept
1      1    Rick    623 2012-01-01        IT
2      2     Dan    515 2013-09-23 Operations
3      3 Michelle    611 2014-11-15        IT
4      4    Ryan    729 2014-05-11        HR
5      5    Gary    843 2015-03-27    Finance
6      6   Rasmi    578 2013-05-21        IT
7      7  Pranab    722 2013-07-30 Operations
8      8   Tusar    633 2014-06-17   Fianance
```

Assignment No : 2.5
Assignment Name : Write program for Creating and Manipulating R Objects in R – List.
Name :
Roll No :

❖ Creating a List

 # Create a list containing strings, numbers, vectors and a logical values.

```
list_data <- list("Red", "Green", c(21,32,11), TRUE, 51.23, 119.1)

print(list_data)
```

OUTPUT:

```
[[1]]
[1] "Red"

[[2]]
[1] "Green"

[[3]]
[1] 21 32 11

[[4]]
[1] TRUE

[[5]]
[1] 51.2

[[6]]
[1] 119
```

❖ Manipulating List Elements

 # Create a list containing a vector, a matrix and a list.

```
list_data <- list(c("Jan","Feb","Mar"), matrix(c(3,9,5,1,-2,8), nrow = 2),
            list("green",12.3))
```

 # Give names to the elements in the list.

```
names(list_data) <- c("1st Quarter", "A_Matrix", "A Inner list")
```

 # Add element at the end of the list.

```
list_data[4] <- "New element"

print(list_data[4])
```

OUTPUT:

```
[[1]]
[1] "New element"
```

 # Remove the last element.

```
list_data[4] <- NULL
```

 # Print the 4th Element.

```
print(list_data[4])
```

```
$<NA>
NULL
```

 # Update the 3rd Element.

```
list_data[3] <- "updated element"

print(list_data[3])
```

```
$`A Inner list`
[1] "updated element"
```

Assignment No : 3
Assignment Name : Write program to demonstrate Loops & Vectorization MissingValues.
Name :
Roll No :

Loops :

1. R repeat loop

```
v <- c("Hello","repeat","loop")
cnt <- 2
repeat
{
        print(v)
        cnt <- cnt+1
        if(cnt > 5)
        {
                break
         }
}
```

OUTPUT :-

```
[1] "Hello"  "repeat" "loop"
[1] "Hello"  "repeat" "loop"
[1] "Hello"  "repeat" "loop"
[1] "Hello"  "repeat" "loop"
```

2. R while loop

```
v <- c("Hello","while loop","example")
cnt <- 2
while (cnt < 7)
{
        print(v)
        cnt= cnt + 1
}
```

OUTPUT :-

```
[1] "Hello"     "while loop" "example"
[1] "Hello"     "while loop" "example"
[1] "Hello"     "while loop" "example"
[1] "Hello"     "while loop" "example"
[1] "Hello"     "while loop" "example"
```

3. R For Loop

```
# Create fruit vector

fruit <- c('Apple', 'Orange',"Guava", 'Pinapple', 'Banana','Grapes')

# Create the for statement

for ( i in fruit)

{

  print(i)

}
```

OUTPUT :-

```
[1] "Apple"
[1] "Orange"
[1] "Guava"
[1] "Pinapple"
[1] "Banana"
[1] "Grapes"
```

Missing values in Vector :-

```
v<-c(1,2,4,NA,5,7,78,8,3,7,NA)

print(v)

v[!is.na(v)]                    #remove NA values from the vector

is.na(v)                        #detecting NA values

which(is.na(v))                  #at which place NA is present

mean(v,na.rm =TRUE)

median(v,na.rm=TRUE)
```

OUTPUT :-

```
[1]  1  2  4 NA  5  7 78  8  3  7 NA
[1]  1  2  4  5  7 78  8  3  7
[1] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
 [9] FALSE FALSE  TRUE
[1]  4 11
[1] 12.77778
[1] 5
```

Assignment No : 4
Assignment Name : Demonstrate Importing and exporting data.
Name :
Roll No :

❖ Imorting Data :-

```
getwd()
setwd("D:/MCA/SEM III/304(C) DA/P_R_Practicals")
```

1. Using read.Xlsx() methods

```
read_xlsx <- read_excel("read.xlsx")
print(read_xlsx)
```

OUTPUT:

	COLOR	TOUGHNESS	FUNGUS	APPEARANCE	POISONOUS
	<chr>	<chr>	<chr>	<chr>	<chr>
1	Green	Hard	NO	Wrinkled	Yes
2	Green	Hard	YES	Smooth	No
3	Brown	Soft	NO	Wrinkled	No
4	Orange	Hard	NO	Wrinkled	Yes
5	Green	Soft	YES	Smooth	Yes
6	Green	Hard	YES	Wrinkled	Yes
7	Orange	Hard	NO	Wrinkled	Yes

2. Using read.csv() methods

```
read_csv <- read.csv("find-s.csv")
print(read_csv)
```

OUTPUT:

	COLOR	TOUGHNESS	FUNGUS	APPEARANCE	POISONOUS
1	Green	Hard	NO	Wrinkled	Yes
2	Green	Hard	YES	Smooth	No
3	Brown	Soft	NO	Wrinkled	No
4	Orange	Hard	NO	Wrinkled	Yes
5	Green	Soft	YES	Smooth	Yes
6	Green	Hard	YES	Wrinkled	Yes

3. Using read.table() methods.

```
x <-read.csv("D:/MCA/SEM III/304(C) DA/P_R_Practicals/find-s.csv",header=TRUE,sep=",")
print(x)
```

OUTPUT:

	COLOR	TOUGHNESS	FUNGUS	APPEARANCE	POISONOUS
1	Green	Hard	NO	Wrinkled	Yes

```
2  Green     Hard    YES     Smooth       No
3  Brown     Soft    NO    Wrinkled       No
4 Orange     Hard     NO   Wrinkled       Yes
5  Green     Soft    YES     Smooth      Yes
6  Green     Hard    YES   Wrinkled       Yes
7 Orange     Hard     NO   Wrinkled       Yes
```

4. **Using read.delim() methods.**

```
x <-read.delim("D:/MCA/SEM III/304(C) DA/P_R_Practicals/find-s.csv",header=TRUE)
print(x)
typeof(x)
```

OUTPUT:

```
COLOR.TOUGHNESS.FUNGUS.APPEARANCE.POISONOUS
1              Green,Hard,NO,Wrinkled,Yes
2               Green,Hard,YES,Smooth,No
3              Brown,Soft,NO,Wrinkled,No
4             Orange,Hard,NO,Wrinkled,Yes
5              Green,Soft,YES,Smooth,Yes
6             Green,Hard,YES,Wrinkled,Yes
7             Orange,Hard,NO,Wrinkled,Yes
> typeof(x)
[1] "list"
```

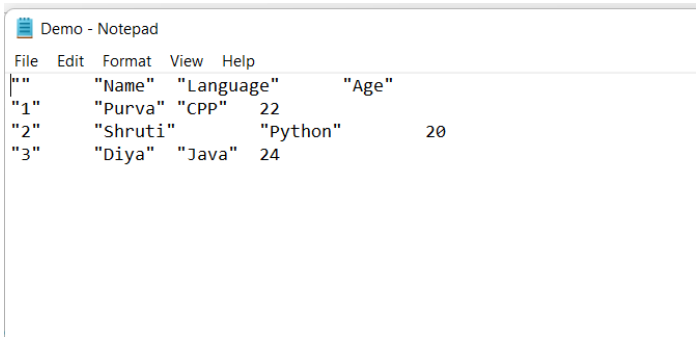
❖ Exporting Data :-

```
getwd()
setwd("D:/MCA/SEM III/304(C) DA/P_R_Practicals")
```

1. **#Export a data frame to a text file using write.table().**

```
df=data.frame(
  "Name"=c("Purva","Shruti","Diya"),
  "Language"=c("CPP","Python","Java"),
  "Age"=c(22,20,24)
)
write.table(df,
          file="Demo.txt",
          sep = "\t",
          row.names = TRUE,
          col.names = NA)
```

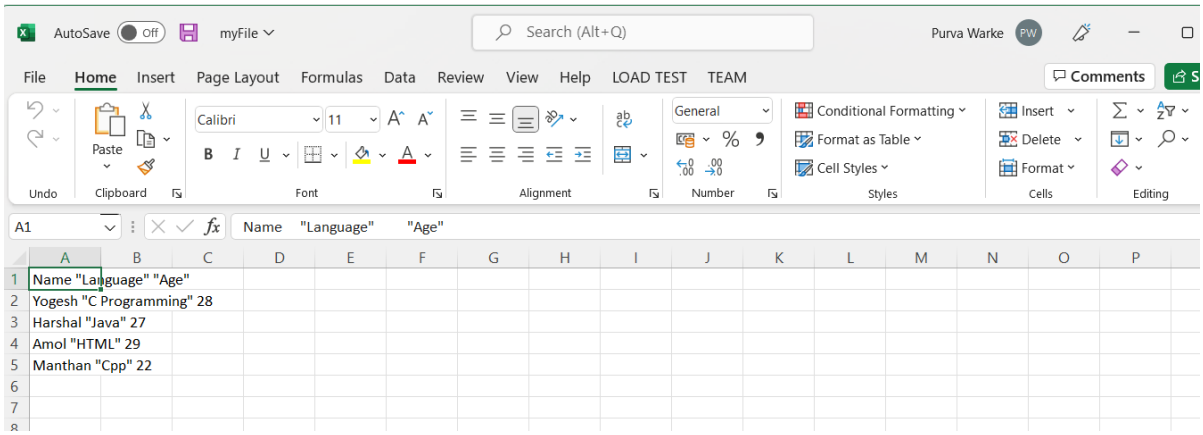
OUTPUT:



2. Exporting Data to a csv file.

```
df=data.frame(
 "Name"=c("Yogesh","Harshal","Amol","Manthan"),
 "Language"=c("C Programming","Java","HTML","Cpp"),
 "Age"=c(28,27,29,22)
)
write.table(df,
        file="myFile.csv",
        sep = "\t ",
        row.names = FALSE)
```

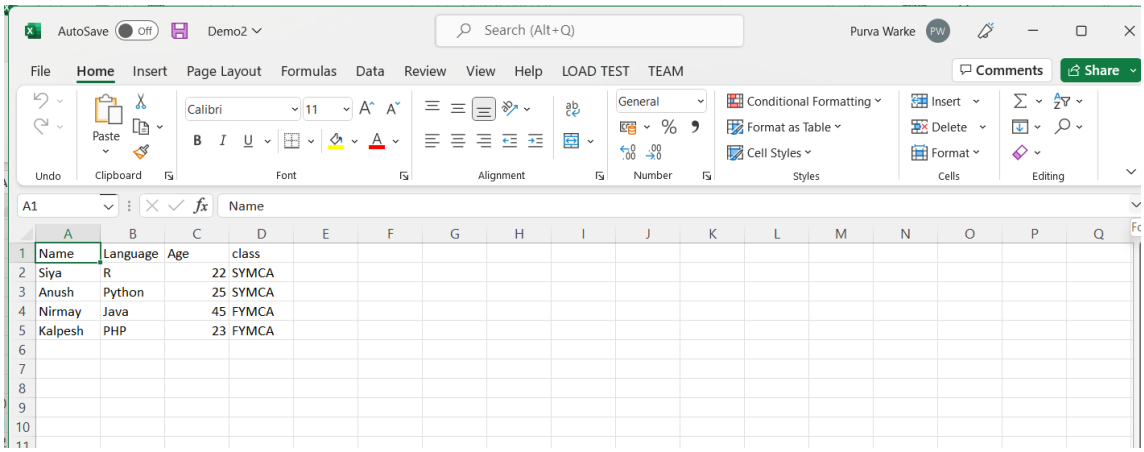
OUTPUT:

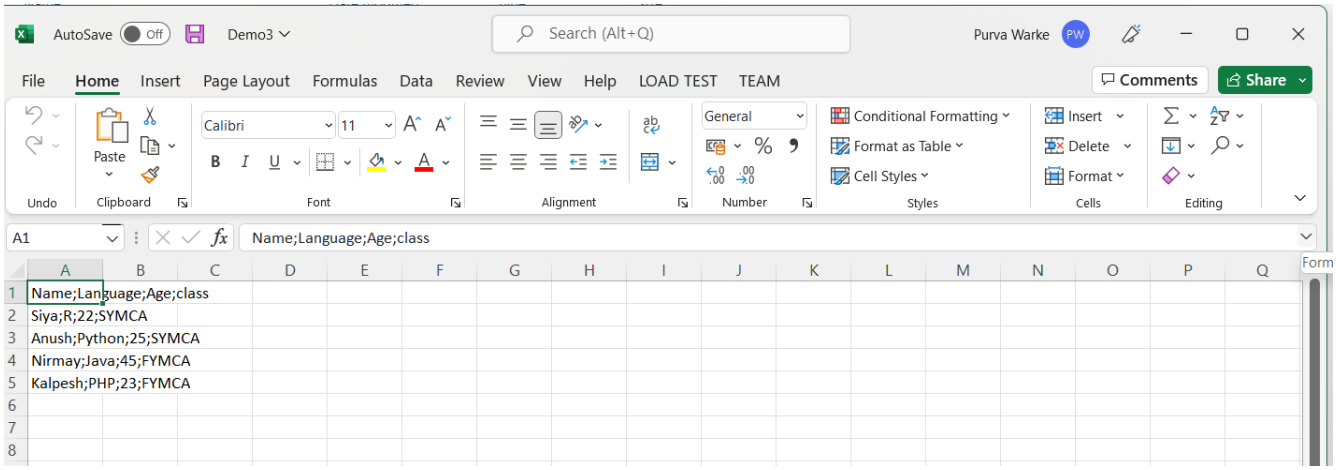


3. Exporting data to a csv2 file

```
library(readr)
df2=data.frame(
 "Name"=c("Siya","Anush","Nirmay","Kalpesh"),
 "Language"=c("R","Python","Java","PHP"),
 "Age"=c(22,25,45,23),
 "class"=c("SYMCA","SYMCA","FYMCA","FYMCA")
)
write_csv(df2,path="Demo2.csv")
write_csv2(df2,path="Demo3.csv")
```

OUTPUT:

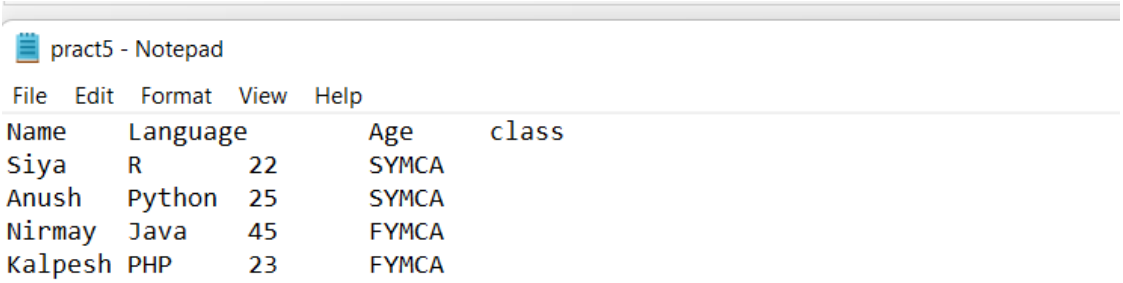




4. Exporting data using write_tsv()function

```
getwd()
#library(readr)
df2=data.frame(
 "Name"=c("Siya","Anush","Nirmay","Kalpesh"),
 "Language"=c("R","Python","Java","PHP"),
 "Age"=c(22,25,45,23),
 "class"=c("SYMCA","SYMCA","FYMCA","FYMCA")
)
write_tsv(df2,path="pract5.txt")
```

OUTPUT:



Assignment No : 5
Assignment Name : Write program for Validating & Exploring Data Manipulations (Summarizing, Sorting, Subsetting, Merging, joining)
Name :
Roll No :

---

1. Summarizing

#create a data frame

```
data1<-data.frame(player=c("A","B","C","D","E"),
                  runs=c(100,220,102,90,60),
                  wickets=c(12,20,9,4,9)
                  )
print(data1)
```

#summarize method

```
library(dplyr)
summarize(data1,sum(runs),mean(runs),mode(wickets))
summarize(data1)
```

OUTPUT :

```
player runs wickets
1      A  100      12
2      B  220      20
3      C  102       9
4      D   90       4
5      E   60       9
sum(runs) mean(runs) mode(wickets)
1       572      114.4       numeric
data frame with 0 columns and 1 row
```

2. Sorting :-

#creating data frame

```
dataBook=data.frame(Clients=c("Diya","Simar","Harshal","Girja","Priti","Kshitij"),
                    Products=c("Prod1","Prod2","Prod3","Prod4","Prod5","Prod6"),
                    Salary=c(6000,5000,8000,4500,3000,9000)
                    )
print(dataBook)
```

#sorting data in ascending order

```
arrange(dataBook,Salary)
```

#sorting data in Descending order

```
dataBook%>%arrange(desc(Salary))
```

OUTPUT :

```
  Clients Products Salary
1    Diya    Prod1   6000
2   Simar    Prod2   5000
3 Harshal    Prod3   8000
4   Girja    Prod4   4500
5   Priti    Prod5   3000
6 Kshitij    Prod6   9000

  Clients Products Salary
1   Priti    Prod5   3000
2   Girja    Prod4   4500
3   Simar    Prod2   5000
4    Diya    Prod1   6000
5 Harshal    Prod3   8000
6 Kshitij    Prod6   9000

  Clients Products Salary
1 Kshitij    Prod6   9000
2 Harshal    Prod3   8000
3    Diya    Prod1   6000
4   Simar    Prod2   5000
5   Girja    Prod4   4500
6   Priti    Prod5   3000
```

3. Subsetting

#subsetting using []Operator

#create vector

```
x<-1.15

cat("Original vector : ",x,"\n")
```

#subsetting vector

```
cat("First 5 values of vector :",x[1:5],"\n ")

cat("without values present at index 1,2 and 3",x[-c(1,2,3)])
```

#subsetting using [[]]Operator

#create list

```
ls<-list(a=1,b=2,c=10,d=20)

cat("Original List : \n")
```

```
print(ls)
```

#select first element of list

```
cat("Element of list : ",ls[[3]],"\n")
```

#subsetting using c function

```
ls2<-list(a=list(x=1,y="Student"),b=1:10)

print(ls2)

cat("Using c function : \n")

print(ls2[[c(1,2)]])

print(ls2[[1]][[2]])
```

#subsetting using $ operator

```
ls3<-list(a="Purva",b=1,c="Hello")

print(ls3)

cat("Using $ operator : \n")

print(ls3$a)
```

OUTPUT :

```
Original vector :  1.15
First 5 values of vector : 1.15 NA NA NA NA
without values present at index 1,2 and 3
Original List :
$a
[1] 1
$b
[1] 2
$c
[1] 10
$d
[1] 20
Using c function :
[1] "Student"
[1] "Student"
$a
[1] "Purva"
$b
[1] 1
$c
[1] "Hello"
Using $ operator :
[1] "Purva"
```

4. Merging :-

#Merge DataFrames by Row Names

```
data_frame1<-data.frame(No=c(1:5),
                        Name=letters[1:5],
                        Salary=c(200,230,600,500,NA)
                        )
print(data_frame1)
data_frame2<-data.frame(No=c(6:8),
                        Name=letters[8:10],
                        Salary=c(500,600,800)
                        )
print(data_frame2)


data_frame_merge<-merge(data_frame1,data_frame2,by="row.names",all=TRUE)
print("Merge Data Frame")
print(data_frame_merge)
```

OUTPUT :

```
  No Name Salary
1  1    a    200
2  2    b    230
3  3    c    600
4  4    d    500
5  5    e     NA
  No Name Salary
1  6    h    500
2  7    i    600
3  8    j    800
[1] "Merge Data Frame"
  Row.names No.x Name.x Salary.x No.y Name.y Salary.y
1         1     1      a      200     6      h     500
2         2     2      b      230     7      i     600
3         3     3      c      600     8      j     800
4         4     4      d      500    NA   <NA>    NA
5         5     5      e       NA    NA   <NA>    NA
```

5. Joining :-

#Using Inner Join

```
data1<-data.frame(ID=c(1:5))
data2<-data.frame(ID=c(4:8))
inner_join(data1,data2,by="ID")
```

#Using Left Join

```
data1<-data.frame(ID=c(1:5),
                  Name=c("Siya","kajal","Riya","Aman","Heena"))
data2<-data.frame(ID=c(4:8),
                  Marks=c(50,60,55,90,85))
left_join(data1,data2,by="ID")
```

OUTPUT :

```
  ID
1  1
2  2
3  3
4  4
5  5
  ID
1  4
2  5
3  6
4  7
5  8
  ID
1  4
2  5
  ID  Name
1  1  Siya
2  2 kajal
3  3  Riya
4  4  Aman
5  5 Heena
  ID Marks
1  4    50
2  5    60
3  6    55
4  7    90
5  8    85
  ID  Name Marks
1  1  Siya    NA
2  2 kajal    NA
3  3  Riya    NA
4  4  Aman    50
5  5 Heena    60
```

Assignment No : 6
Assignment Name : Write program to implement the following analysis techniques using R ,Statistical hypothesis generation and testing ,Chi-Square test, t-Test, Correlation analysis
Name:
Roll No :

#statistical Hypothesis testing

#one-sample T-testing

```
x<-rnorm(100)#sample vector

t.test(x,mu=5)#one sample test
```

OUTPUT:

```
One Sample t-test
data:  x
t = -52.268, df = 99, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 5
95 percent confidence interval:
 -0.2733968  0.1123422
sample estimates:
 mean of x
-0.0805273
```

#two-sample T-testing

```
x<-rnorm(100)

y<-rnorm(100)

t.test(x,y)
```

OUTPUT:

```
Welch Two Sample t-test
data:  x and y
t = -0.056338, df = 197.95, p-value = 0.9551
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.2814629  0.2658276
sample estimates:
  mean of x   mean of y
-0.04196545 -0.03414780
```

#directional Hypothesis

```
t.test(x,mu=2,alternative = 'greater')
```

OUTPUT:

```
One Sample t-test
```

```
data:  x
t = -20.982, df = 99, p-value = 1
alternative hypothesis: true mean is greater than 2
95 percent confidence interval:
 -0.2035555       Inf
sample estimates:
  mean of x
-0.04196545
```

#one-sample u-test

```
wilcox.test(y,exact=FALSE)
```

OUTPUT:

```
Wilcoxon signed rank test with continuity
       correction
data:  y
V = 2373, p-value = 0.6024
alternative hypothesis: true location is not equal to 0
```

#two-sample u-test

```
wilcox.test(x,y)
```

OUTPUT:

```
Wilcoxon rank sum test with continuity
       correction
data:  x and y
W = 4878, p-value = 0.7666
alternative hypothesis: true location shift is not equal to 0
```

#correlation Test

```
cor.test(matcars$mpg,matcars$hp)
```

OUTPUT:

```
Pearson's product-moment correlation
data:  mtcars$mpg and mtcars$hp
t = -6.7424, df = 30, p-value = 1.788e-07
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.8852686 -0.5860994
sample estimates:
      cor
-0.7761684
```

#Chi-Square Test

```
library(MASS)
```

#create Dataframes

```
print(str(survey))
```

OUTPUT:

```
'data.frame':237 obs. of  12 variables:
 $ Sex   : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 1 2 1 2 2 ...
 $ Wr.Hnd: num  18.5 19.5 18 18.8 20 18 17.7 17 20 18.5 ...
 $ NW.Hnd: num  18 20.5 13.3 18.9 20 17.7 17.7 17.3 19.5 18.5 ...
 $ W.Hnd : Factor w/ 2 levels "Left","Right": 2 1 2 2 2 2 2 2 2 2 ...
 $ Fold  : Factor w/ 3 levels "L on R","Neither",..: 3 3 1 3 2 1 1 3 3 3 ...
 $ Pulse : int  92 104 87 NA 35 64 83 74 72 90 ...
 $ Clap  : Factor w/ 3 levels "Left","Neither",..: 1 1 2 2 3 3 3 3 3 3 ...
 $ Exer  : Factor w/ 3 levels "Freq","None",..: 3 2 2 2 3 3 1 1 3 3 ...
 $ Smoke : Factor w/ 4 levels "Heavy","Never",..: 2 4 3 2 2 2 2 2 2 2 ...
 $ Height: num  173 178 NA 160 165 ...
 $ M.I   : Factor w/ 2 levels "Imperial","Metric": 2 1 NA 2 2 1 1 2 2 2 ...
 $ Age   : num  18.2 17.6 16.9 20.3 23.7 ...
NULL
```

#create a data frame from the main data set

```
stu_data=data.frame(survey$Smoke,survey$Exer)
```

#create a contingency table with the needed variables

```
stu_data=table(survey$Smoke,survey$Exer)

print(stu_data)
```

OUTPUT:

```
       Freq None Some
  Heavy    7    1    3
  Never   87   18   84
  Occas   12    3    4
  Regul    9    1    7
```