EXPERIMENT 1

```
# print hello world
str <- "hello world!!"
str

# various ways to declare variable
variable.1 = c(1,2,3)
variable.2 <- c("lotus" , "rose")
c(FALSE , 1) -> variable.3
variable.1
cat("variable.1 is " , variable.1, "\n")
cat("variable.2 is " , variable.2, "\n")
cat("variable.3 is " , variable.3, "\n")

# perform arithmetic operations
a <- c(10,20,30,40)
b <- c(2,2,4,3)
cat("sum= ", (a+b), "\n")
cat("difference= ", (a-b), "\n")
cat("product= ", (a*b), "\n")
cat("quotient= ", (a/b), "\n")
cat("remainder= ", (a%%b), "\n")
cat("integer division= ", (a%/%b), "\n")
cat("exponention = ", (a^b), "\n")

# perform relational operators
a <- c(10,20,30,40)
b <- c(4,5,90,7)
cat(a , " less than " , b , (a < b) , "\n" )
cat(a , " greater than " , b , (a > b) , "\n" )
cat(a , " less than or equal to " , b , (a <= b) , "\n" )
cat(a , " greater than or equal to" , b , (a >= b) , "\n" )
cat(a , " equal to " , b , (a == b) , "\n" )
cat(a , " not equal to " , b , (a != b) , "\n" )
```

```
# perform logical operations
a <- c(10,20,30,40)
b <- c(2,2,8,5)
cat(a,"logical NOT" , (!a) , "\n")
cat(a,"element wise logical AND" , b,(a&b) , "\n")
cat(a,"element wise logical OR" , b , (a|b) , "\n")

# perform assignment operators
var.a=c(0,20,TRUE)
var.b <- c(0,20,TRUE)
var.c <<- c(0,20,TRUE)
var.a
var.b
var.c
c(1,2,TRUE) ->v1
c(1,2,TRUE) -> v2
v1
v2

# Demonstrate numeric DataType
x = 10.9
x
y=5
y
class(x)
class(y)
is.integer(x)
is.integer(y)

# demonstrate integer data type
x = as.integer(3)
x
class(x)
is.integer(x)
y = as.integer(3.23)
```

```
y
z=as.integer("7.81")
z
as.integer(TRUE)
as.integer(FALSE)

# demonstrate complex data types
x=5+4i
x
class(x)
is.complex(x)
y=as.complex(3)
y

# demonstrate logical data types
x=1;y=2
z=x>y
z
class(z)
as.logical(1)
as.logical(0)

# demonstrate character data types
x="abc"
y=as.character(7.8)
x
y
class(y)
```

Experiment 2

```
#Creating vector by using colon(:) operator.

x <- 1:7

x

x <- -2.5:10.5

x


#Creating vector by using c() function.

x <- c(10,2.5,TRUE)

x

typeof(x)

x <- c(1,2.8,TRUE,"Apple")

x

typeof(x)

length(x)


#Creating vector by using sequence(seq) function.

x <- seq(2,3, by=0.2)

x

x <- seq(2,3, length=3)

x


#Combining vectors

n =c (2,3,4)

m = c("a","b","c","d")

c(n,m)


#Accessing vector element by using Integer vector as index.

s = c("a","b","c","d","e")

s[3]

s[10]


s = c("a","b","c","d","e")

s[c(2,3)]

s[c(2,1,3)]
```

```
s[2:4]

#Accessing vector element by using logical vector as index.
s = c(1,2,3,-5,-6)
s[c(TRUE,FALSE,FALSE,TRUE,FALSE)]
TRUE
s[s<0]
s[s>0]

#Accessing vector element by using character vector as index.
v = c("Jack","Joe","Tom")
v
names(v) = c("first","second","third")
v["second"]
v[c("third","first","second")]

#Modifying vectors.
x =c(10,20,30,40,50,60)
x
x[2] <- 90
x
x <- x[1:4]
x

#Deleting vectors.
x =c(10,20,30,40,50,60)
x
x <- NULL
x

#Vector arithmetic and recycling.
x =c(10,20,30)
x
y =c(1,2,3)
x+y
```

```
x+1
y+c(4,5,5)

#Vector element sorting.
x = c(7,1,8,4,2,7,4,6,2,2,4)
sort(x)
sort(x,decreasing = TRUE)
x
flowers = c("lotus","rose","jasmine","daisy","lilly")
sort(flowers)
sort(flowers,decreasing = TRUE)
flowers

#Reading vectors.
my.name <- readline(prompt = "Enter name:")
my.age <- readline(prompt = "Enter age:")
my.bool <- readline(prompt = ("Enter (TRUE/FALSE): "))
my.age <- as.integer(my.age)
my.bool <- as.logical(my.bool)
my.name
my.age
my.bool

#Creating Lists.
n = list(c(2,3,5),c("a","b","c","d","e"),c("TRUE,FALSE,TRUE,FALSE,FALSE"),3)
n

#Creating matrices.
m <- matrix(c(1:12),nrow=4,byrow=TRUE)
m
n <- matrix(c(1:12),nrow=4,byrow=FALSE)
n
rnames =c("r1","r2","r3","r4")
cnames =c("c1","c2","c3")
p <- matrix(c(1:12),nrow=4,byrow=TRUE, dimnames=list(rnames,cnames))
```

```
p


#Creating matrices by using cbind() and rbind().
m = cbind(c(1,2,3),c(4,5,6))
m
n = rbind(c(1,2,3),c(4,5,6))
n


#Creating factors.
x <- factor(c("single","married","married","single","divorced"))
x
class(x)
levels(x)
str(x)


#Creating data frames.
x <- data.frame("roll"=1:2,"name"=c("Aishwarya","Sakshi"),"age"=c(20,21))
x
names(x)
nrow(x)
ncol(x)
str(x)
summary(x)
```

EXPERIMENT 3

```
# random sampling
sample(1:20,10)


# random sampling with and without replacement
sample(1:6,4, replace = TRUE)
sample(1:6,4, replace = FALSE)
```

```
# Random sampling for character.
LETTERS
sample(LETTERS)
sample(LETTERS)

# Random sampling on vector without replacement.
data <- c(1,3,5,6,7,8,10,11,12,14)
sample(x=data , size=5)

# Random sampling on vector with replacement.
data <- c(1,3,5,6,7,8,10,11,12,14)
sample(x=data , size=5 , replace=TRUE)

# Random sampling on data frames.
df<-data.frame(x=c(3,5,6,6,8,12,14) , y=c(12,6,4,23,25,8,9) , z=c(2,7,8,8,15,17,29))
df
rand_df<-df[sample(nrow(df), size =3),]
rand_df


# stratified sampling on data frames using number of rows
install.packages("dplyr")
library(dplyr)
set.seed(1)
df<- data.frame(grade=rep(c('Freshman','Sophomore','Junior','Senior') , each=100), gpa=rnorm(400,mean =
85 , sd=3))
head(df)
strat_sample<-df%>%
group_by(grade) %>%
sample_n(size=10)

table(strat_sample$grade)

# stratifies sampling on data frames using fraction of rows
library(dplyr)
```

```
strat_sample<-df%>%
  group_by(grade) %>%
  sample_frac(size=.15)
table(strat_sample$grade)
```

EXPERIMENT 4

```
# to calculate mean of a vector
marks <- c(97,78,57,64,87)
# calculate average marks
result <- mean(marks)
print(result)

# to calculate median on a vector
marks <- c(97,78,57,64,87)
result <- median(marks)
print(result)

# calculate mode
marks <- c(97,78,57,78,97,66,87,64,87,78)
mode = function(){
  return(names(sort(-table(marks)))[1])
}
mode()


# calculate central tendency on imported files
myData = read.csv("CardioGoodFitness.csv")
print(head(myData))

# calculate mean on imported files
myData = read.csv("CardioGoodFitness.csv")
mean = mean(myData$Age)
```

```
print(mean)

# calculate median
median = median(myData$Age)
print(median)

# calculate mode
mode = function(){
  return(sort(-table(myData$Age))[1])
}
mode()

# calculate mode using moodest library
install.packages("modeest")
library(modeest)
myData = read.csv("CardioGoodFitness.csv")
mode = mfv(myData$Age)
print(mode)
```

EXPERIMENT 5

```
# calculate range of vectors
x <- c(5,5,8,12,15,16)
print(range(x))

# using max() and min() function
print(max(x) -min(x))

# to calculate standard deviation on vector
x <- c(5,5,8,12,15,16)
d <- sqrt(var(x))
print(d)

# calculate variance
```

```
x <- c(5,5,8,12,15,16)
print(var(x))

# calculate percentile
x <- c(5,5,8,12,15,16)
res <- quantile(x, probs = 0.5)
res

# calculate interquartile range on vectors
x <- c(5,5,8,12,15,16)
print(IQR(x))

# calculate variability on imported files
myData = read.csv("CardioGoodFitness.csv")
print(head(myData))

# calculate range
print(range(myData$Miles))

print(max(myData$Miles) -min(myData$Miles))

# calculate SD
print(sqrt(var(myData$Miles)))

print(sd(myData$Miles))

# calculate variance
print(var(myData$Miles))

# calculate percentile
print(quantile(myData$Miles , probs = 0.5))
```

EXPERIMENT 6

```
# Horizontal Bar plot for
# ozone concentration in air
barplot(airquality$Ozone,
        main = 'Ozone Concentration in air',
        xlab = 'ozone levels' , horiz = TRUE)

# vertical Bar Plot for
# Ozone Concentration in air
barplot(airquality$Ozone, main = 'Ozone Concentration in air' ,
        xlab = 'ozone levels' , col = 'blue' , horiz = FALSE)




data(airquality)
hist(airquality$Temp , main="La Guardia Airport's\
Maximum Temperature(Daily)",
    xlab="Temperature(Farenheit)",
    xlim=c(50,125),col="green")

# Box plot for average wind speed
data(airquality)
boxplot(airquality$Wind , main="Average Wind Speed\
at La Guarda Airport " ,
        xlab="Miles Per Hour" , yabl="wind" ,
        col="yellow" , border = "red" ,
        horizontal = TRUE , notch = TRUE)



# Multiple box plots , each representing
# an air quality parameter
boxplot(airquality[,0:4],
        main = 'Box Plots for Air Quality parameters')
```

```
# scatter plots for ozone concentration per month
data("airquality")
plot(airquality$Ozone , airquality$Month,
    main="scatterplot example" ,
    xlab = "ozone concentration in parts per billion" ,
    ylab ="Month of observation" , pch=19)



# set seed for reproducibility
# set.seed(110)
# create example data
data <- matrix(rnorm(50,0,5), nrow =5 , ncol=5)
#column names
colnames(data) <- paste0("col",1:5)
rownames(data) <- paste0("row" , 1:5)
# draw a heatmap
heatmap(data)



# adding titles and labelling axes to plot
cone <- function(x,y){
  sqrt(x^2+y^2)
}
#prepare variables
x <- y <- seq(-1,1, length = 30)
z <- outer(x,y,cone)
# plot the 3D surface
# adding titlles and labelling axes to plot
persp(x,y,z,
     main="perspective plot of a cone" ,
     zlab = "height" ,
     theta = 30 , phi=15,
     col="lightgreen" , shade=0.4)
```

EXPERIMENT 7

date()

```
# get current date
Sys.Date()
Sys.time()

# get current date using lubricate package
install.packages('lubridate')
library(lubridate)
now()

# extract years,months , days from multiple date values in r
library(lubridate)
dates <- c("2022-07-11" , "2012-04-19" , "2017-03-08")
year(dates)
month(dates)
mday(dates)

# manipulate date values in R
my_date <- as.Date("2004-01-22")
my_date
class(my_date)
format(my_date , "%d-%m-%Y")
format(my_date , "%Y-%h-%d")
format(my_date , "%Y-%m-%d-%H-%M-%S")
format(my_date , "%Y/%m/%d")


# update()
date <- ymd("2004-01-22")
update(date,year = 2008 , month = 2 , mday = 1)
update(date,minute =10,second=2)
```