NAME: Hammad Ansari **ROLL NO: 2018450002**

Experiment No. 2

Title: To use control structures, functions and Data Manipulation in R

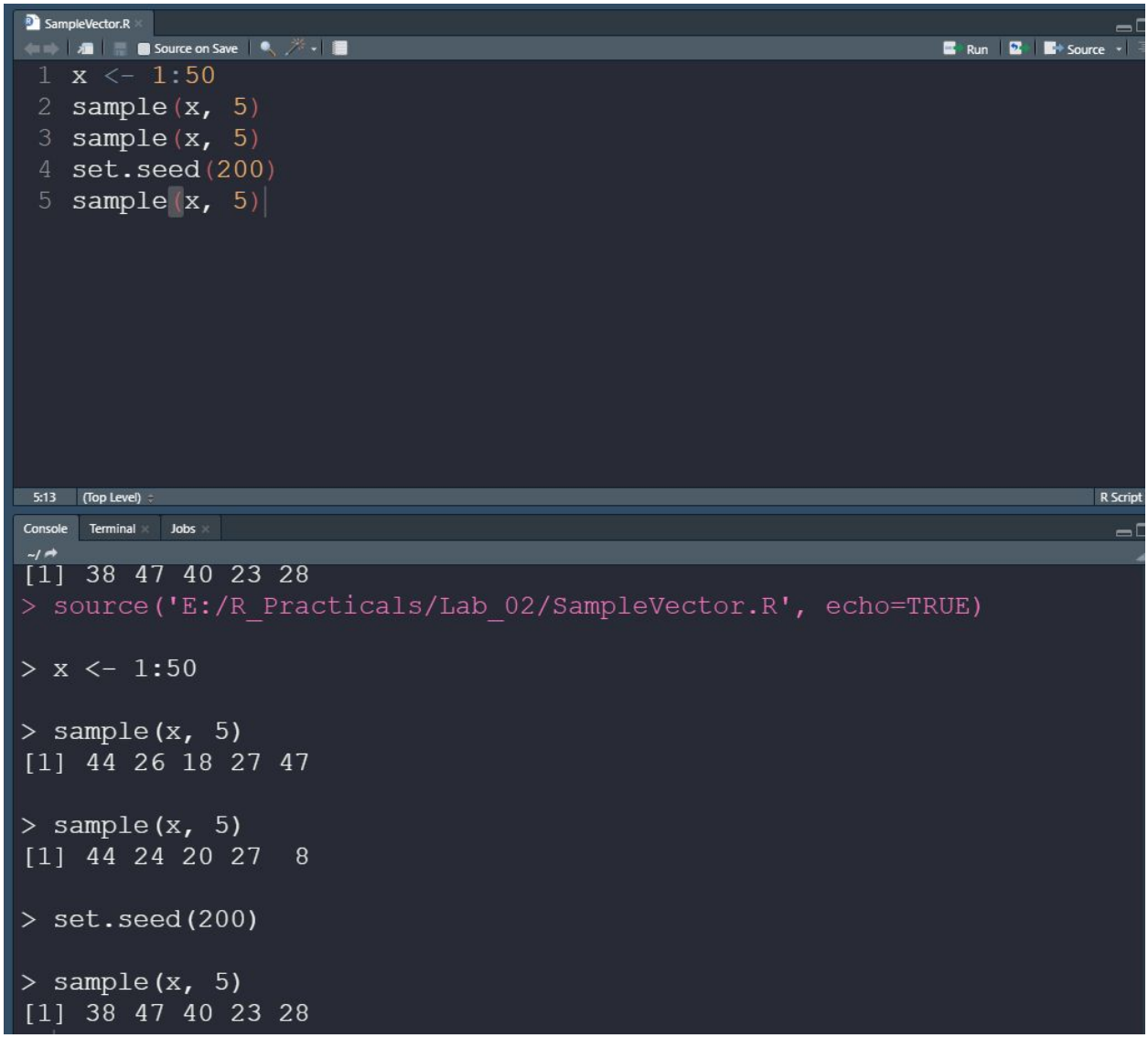
List of Programs:

1. R Program to take Sample from a Population, simulate tossing a coin 10 times experiment.

   Code:

   ```
   x <- 1:50

   sample(x, 5)

   sample(x, 5)

   set.seed(200)

   sample(x, 5)
   ```

   Screenshot:



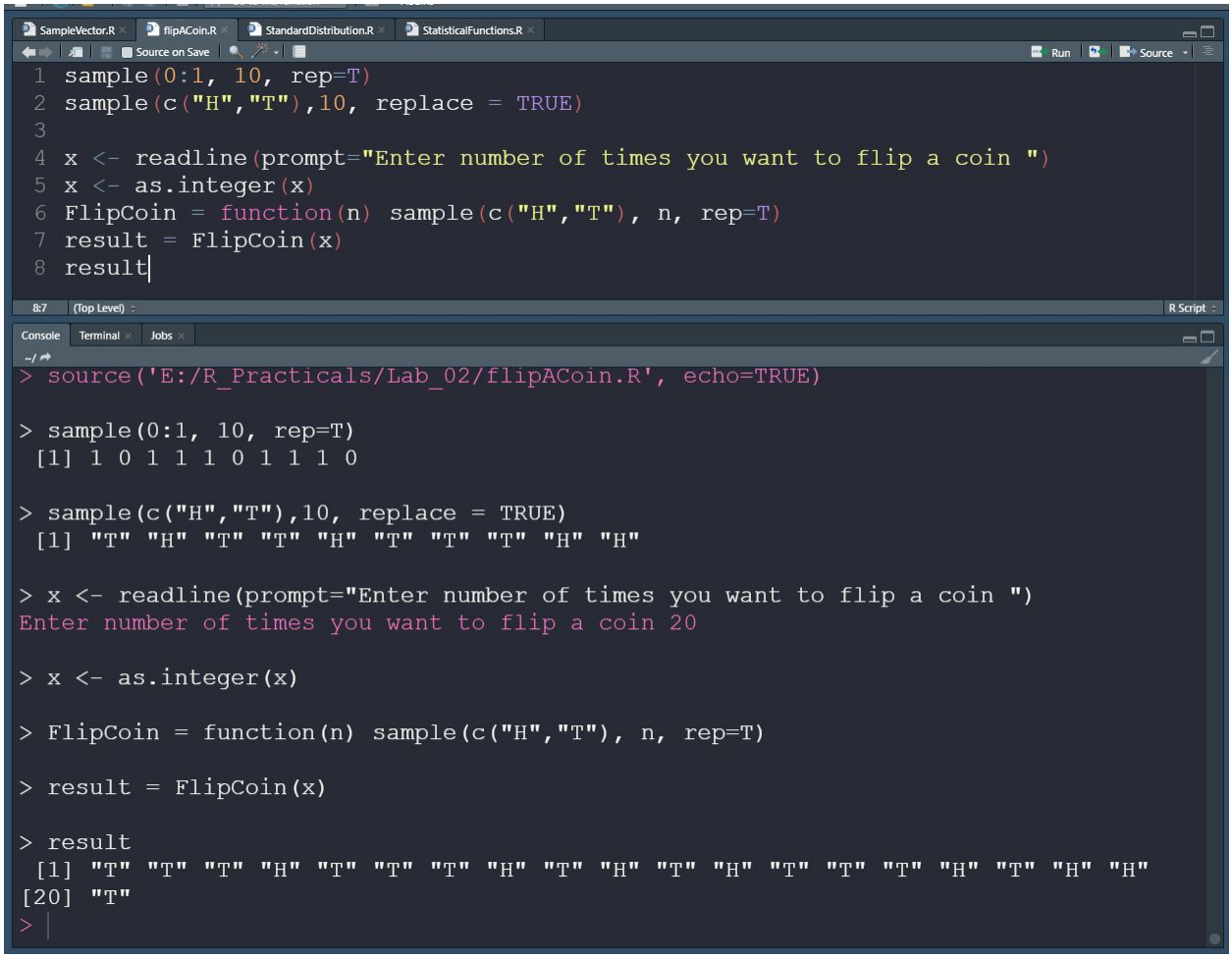
Code:

```
sample(0:1, 10, rep=T)

sample(c("H","T"),10, replace = TRUE)


x <- readline(prompt="Enter number of times you want to flip a coin ")

x <- as.integer(x)

FlipCoin = function(n) sample(c("H","T"), n, rep=T)
```

result = FlipCoin(x)

result

Screenshot:



2. Apply statistical function on a vector to find min, max, variance, sum, median, sd, range, summary, etc.

Code:

v <- 1:1000 #data set for speed of cars

speed <- sample(v, 500)

min(speed) #Min

max(speed) #Max

sum(speed) #Sum

range(speed) #speed between range

mean(speed) #Mean

median(speed) #Median

var(speed) #Variance

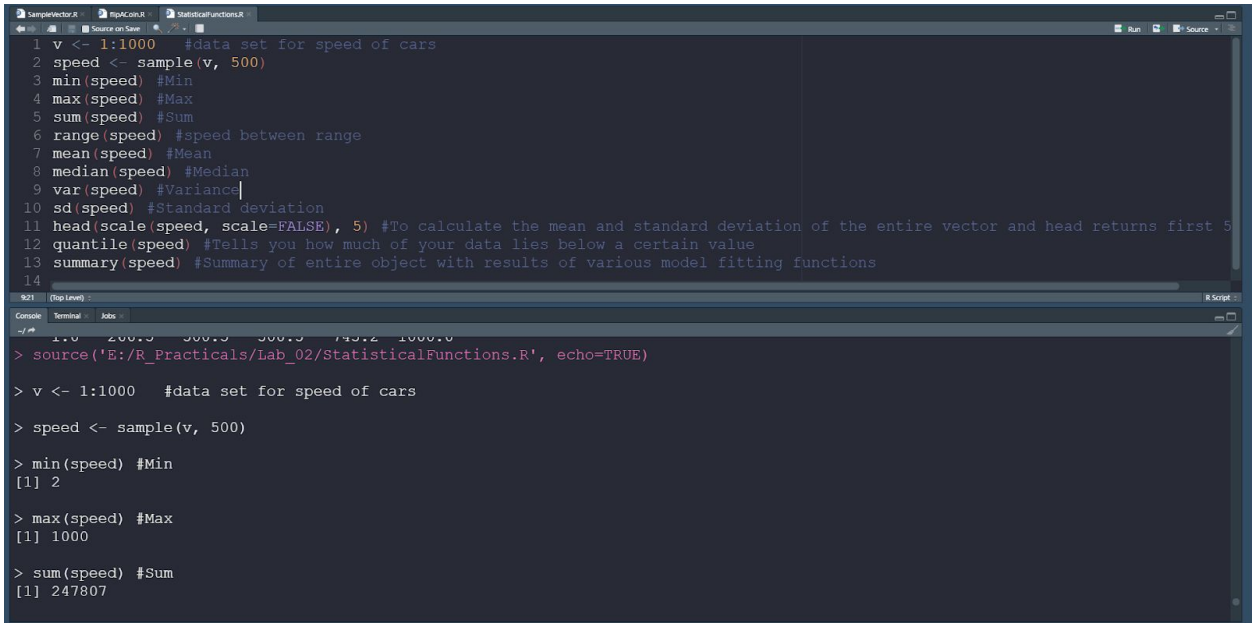
sd(speed) #Standard deviation

head(scale(speed, scale=FALSE), 5) #To calculate the mean and standard deviation of the entire vector and head returns first 5 rows

quantile(speed) #Tells you how much of your data lies below a certain value

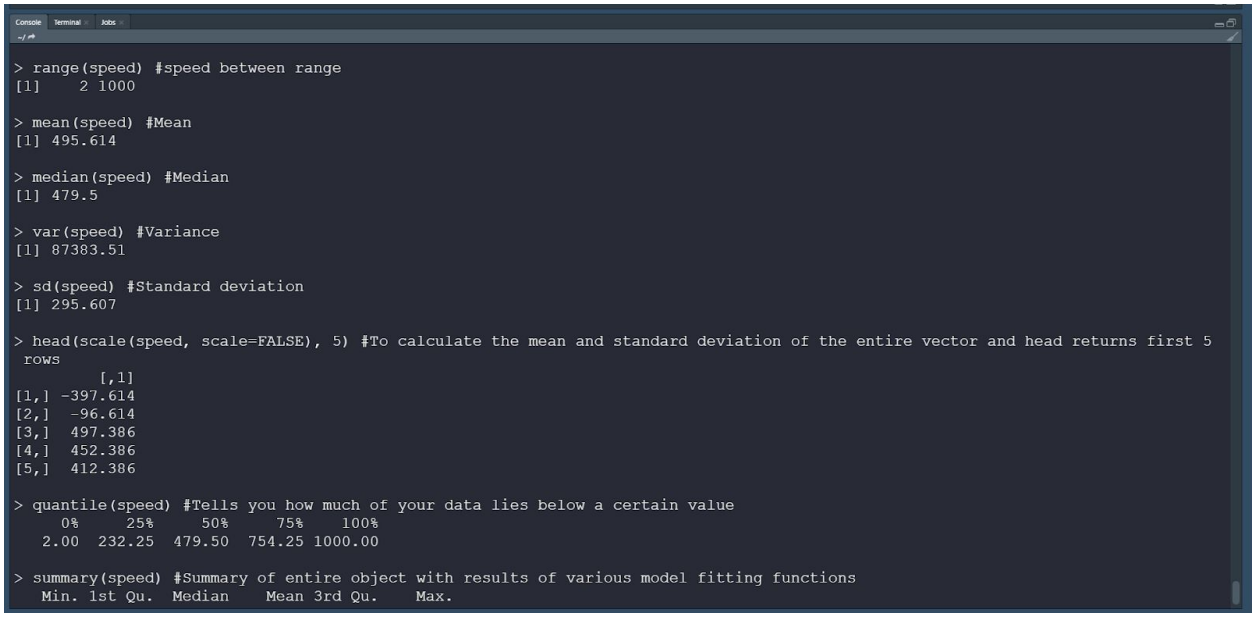
summary(speed) #Summary of entire object with results of various model fitting functions

Screenshot:



```
 1 v <- 1:1000   #data set for speed of cars
 2 speed <- sample(v, 500)
 3 min(speed) #Min
 4 max(speed) #Max
 5 sum(speed) #Sum
 6 range(speed) #speed between range
 7 mean(speed) #Mean
 8 median(speed) #Median
 9 var(speed) #Variance
10 sd(speed) #Standard deviation
11 head(scale(speed, scale=FALSE), 5) #To calculate the mean and standard deviation of the entire vector and head returns first 5
12 quantile(speed) #Tells you how much of your data lies below a certain value
13 summary(speed) #Summary of entire object with results of various model fitting functions
14
```

```
> source('E:/R_Practicals/Lab_02/StatisticalFunctions.R', echo=TRUE)

> v <- 1:1000   #data set for speed of cars

> speed <- sample(v, 500)

> min(speed) #Min
[1] 2

> max(speed) #Max
[1] 1000

> sum(speed) #Sum
[1] 247807
```



```
> range(speed) #speed between range
[1]    2 1000

> mean(speed) #Mean
[1] 495.614

> median(speed) #Median
[1] 479.5

> var(speed) #Variance
[1] 87383.51

> sd(speed) #Standard deviation
[1] 295.607

> head(scale(speed, scale=FALSE), 5) #To calculate the mean and standard deviation of the entire vector and head returns first 5
 rows
         [,1]
[1,] -397.614
[2,]  -96.614
[3,]  497.386
[4,]  452.386
[5,]  412.386

> quantile(speed) #Tells you how much of your data lies below a certain value
     0%     25%     50%     75%    100%
   2.00  232.25  479.50  754.25 1000.00

> summary(speed) #Summary of entire object with results of various model fitting functions
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

3. Generate Random Number from Standard Distributions[Uniform and Normal].

   Code:

   # Create a sample of 50 numbers which are normally distributed.

   normalVector <- 1:1000

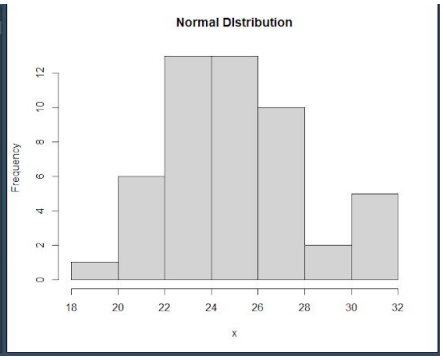
   observations <- sample(normalVector, 50);

```
y <- runif(observations, min=2, max=49)    # define the range between 2 and 49

x <- rnorm(observations, mean=25, sd=3)    # provide our own mean and
standard deviation


hist(x, main = "Normal DIstribution")

hist(y, main = "Uniform DIstribution")


# Create a sequence of numbers between -10 and 10 increment by 0.1.

x <- seq(-10, 10, by = .1)


# Choose the mean as 2.5 and standard deviation as 0.5.

y <- dnorm(x, mean = 2.5, sd = 0.5) # Gives height of the probability distribution

plot(x, y, main = "Height of Probability Dist")

y <- pnorm(x, mean = 2.5, sd = 0.5) # Gives the probability of a normally
distributed random number to be less that the value of a given number. It is also
called "Cumulative Distribution Function"

plot(x, y, main = "Cumulative Distribution")

y <- qnorm(x, mean = 2.5, sd = 0.5) # Gives a number whose cumulative value
matches the probability value

plot(x, y, main = "Cumulative = Probability")
```

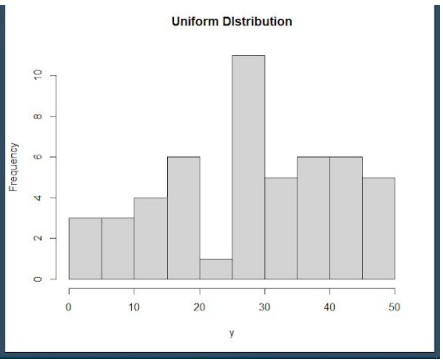
Screenshot:

NAME: Hammad Ansari **ROLL NO: 2018450002**

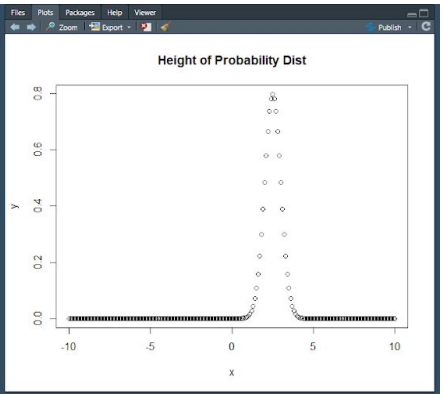
```
> hist(x, main = "Normal DIstribution")

> hist(y, main = "Uniform DIstribution")
> source('E:/R_Practicals/Lab_02/StandardDistribution.R', echo=TRUE)

> # Create a sample of 1000 numbers which are normally distributed.
> normalVector <- 1:1000

> observations <- sample(normalVector, 50);

> y <- runif(observations, min=2, max=49)     # define the range between 2 and 49

> x <- rnorm(observations, mean=25, sd=3)     # provide our own mean and standard dev
iation

> hist(x, main = "Normal DIstribution")
>
```



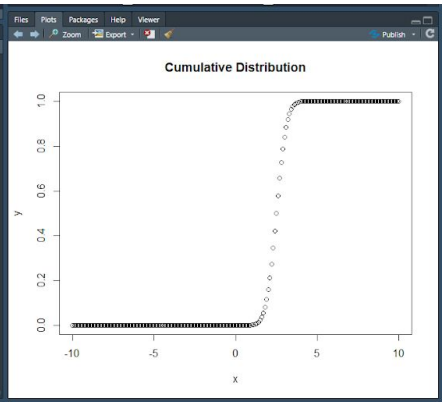

```
> hist(y, main = "Uniform DIstribution")
> source('E:/R_Practicals/Lab_02/StandardDistribution.R', echo=TRUE)

> # Create a sample of 50 numbers which are normally distributed.
> normalVector <- 1:1000

> observations <- sample(normalVector, 50);

> y <- runif(observations, min=2, max=49)     # define the range between 2 and 49

> x <- rnorm(observations, mean=25, sd=3)     # provide our own mean and standard dev
iation

> hist(x, main = "Normal DIstribution")

> hist(y, main = "Uniform DIstribution")
>
```

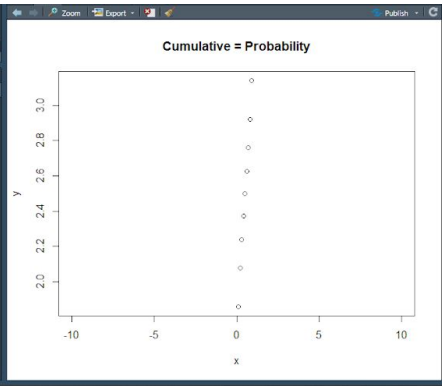



```
> x <- seq(-10, 10, by = .1)

> # Choose the mean as 2.5 and standard deviation as 0.5.
> y <- dnorm(x, mean = 2.5, sd = 0.5) # Gives height of the probability distribution

> plot(x, y, main = "Height of Probability Dist")

> y <- pnorm(x, mean = 2.5, sd = 0.5) # Gives the probability of a normally distribu
ted random number to be less that the value of a given number. It  .... [TRUNCATED]

> plot(x, y, main = "Cumulative Distribution")

> y <- qnorm(x, mean = 2.5, sd = 0.5) # Gives a number whose cumulative value matche
s the probability value

> plot(x, y, main = "Cumulative = Probability")
Warning message:
In qnorm(x, mean = 2.5, sd = 0.5) : NaNs produced
>
```



```
> x <- seq(-10, 10, by = .1)

> # Choose the mean as 2.5 and standard deviation as 0.5.
> y <- dnorm(x, mean = 2.5, sd = 0.5) # Gives height of the probability distribution

> plot(x, y, main = "Height of Probability Dist")

> y <- pnorm(x, mean = 2.5, sd = 0.5) # Gives the probability of a normally distribu
ted random number to be less that the value of a given number. It  .... [TRUNCATED]

> plot(x, y, main = "Cumulative Distribution")

> y <- qnorm(x, mean = 2.5, sd = 0.5) # Gives a number whose cumulative value matche
s the probability value

> plot(x, y, main = "Cumulative = Probability")
Warning message:
In qnorm(x, mean = 2.5, sd = 0.5) : NaNs produced
>
```





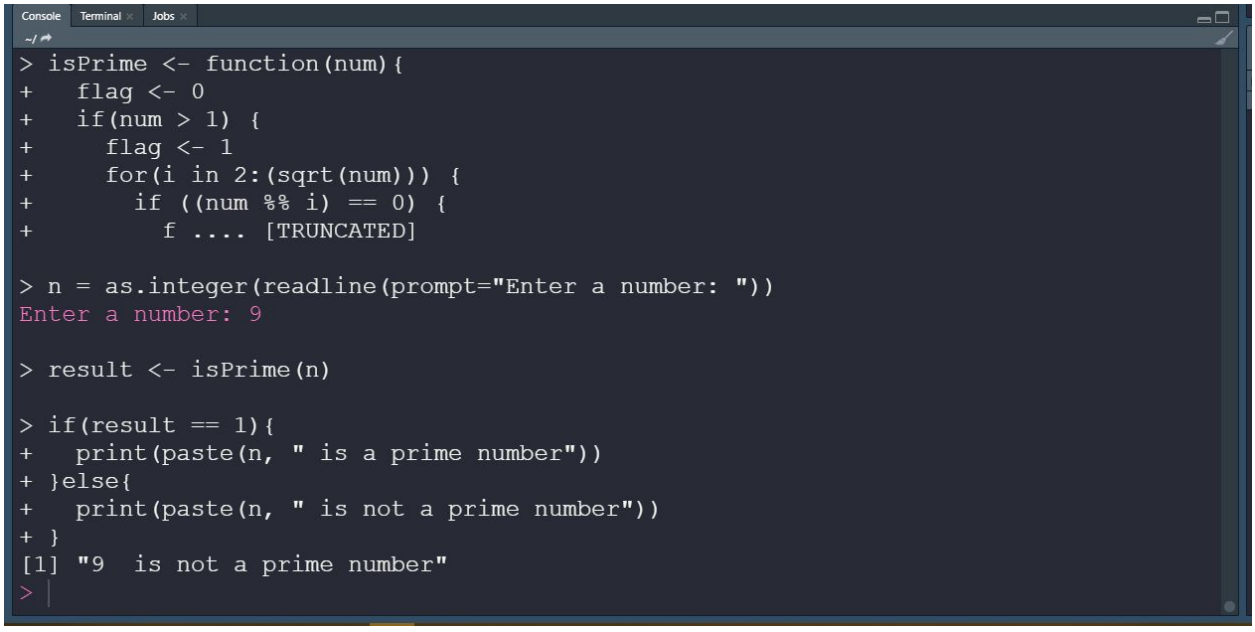
4. Check whether the Number is Prime or not.

Code:

```
isPrime <- function(num){
 flag <- 0
 if(num > 1) {
  flag <- 1
  for(i in 2:(sqrt(num))) {
   if ((num %% i) == 0) {
    flag <- 0
    break
   }
```

```
   }
  }
  if(num == 2)    flag <- 1
  return (flag)
}
n = as.integer(readline(prompt="Enter a number: "))
result <- isPrime(n)
if(result == 1){
  print(paste(n, " is a prime number"))
}else{
  print(paste(n, " is not a prime number"))
}
```

Screenshot:



5. Use the “dplyr” package to perform data manipulation operations on any dataset:

   Use functions like select, filter, arrange, mutate, summarise, group_by.

   Code:

```
#if you are having difficulties while installing please run r studio as administrator

#install.packages("dplyr") uncomment this to install


library("dplyr")

library("datasets")

data(mtcars)

summary(mtcars)

head(mtcars)


#To select the following columns and (-) to hide the column

s <- select(mtcars, hp, mpg, cyl, -am)

head(s)


#To select columns with numeric indexes

s2 <- select(mtcars,c(3:5))

head(s2)
```

```
#To filter automatic and manual transmission cars 1 is for automatic and 0 for manual

f <- filter(mtcars, am == 1 )

head(f)


#To filter cars with 6 or more cylinders and automatic transmission

f2 <- filter(mtcars, cyl >= 6, am==1)

head(f2)


#filter with selected elements

f3 <- filter(s, cyl == 8)

print(f3)


#To create a column “More Power” which stores TRUE if given condition is TRUE

newCol <- mutate(mtcars, MorePower = (hp >= 200))

head(newCol)


#To check how many cars satisfy condition

table(newCol$MorePower)


#To arrange miles per gallon in ascending order

arr <- arrange(newCol, mpg)
```

```
head(arr)


#To arrange miles per gallon in descending order
arr2 <- arrange(newCol, desc(mpg))
head(arr2)


#To arrange cars in ascending  order by miles per gallon
head(mtcars[order(mtcars$mpg), ])


#To arrange cars in descending  order by miles per gallon
head(mtcars[order(mtcars$mpg, decreasing = TRUE), ])


#To arrange cars in order with cylinder, v shape and miles per gallon
mtcars[with(mtcars, order(cyl, vs, mpg)), ]


# summarize to find out (mean, median, mode, etc.)
summarised <- summarise(arr, Mean.HorsePower = mean(hp))
head(summarised)


#To find mean horse power by cylinder, we use grouping as follows
gp1 <- group_by(mtcars, cyl)
grouped <- summarise(gp1, Mean.HorsePower = mean(hp))
```

```
head(grouped)


#Pipe operator

#To get rows with the following conditions

mtcars %>% filter(am == 1, hp > 200)


#To find mean miles per gallon by automatic transmission

mtcars  %>% group_by(gear) %>% summarise(Mean.MPG = mean(mpg))
```

Output:

```
> #if you are having difficulties while installing please run rstudio as
administrator

> #install.packages("dplyr") uncomment this to install

> .... [TRUNCATED]


> library("datasets")


> data(mtcars)


> summary(mtcars)
      mpg             cyl             disp             hp             drat
 Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0   Min.   :2.760
 1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5   1st Qu.:3.080
 Median :19.20   Median :6.000   Median :196.3   Median :123.0   Median :3.695
 Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7   Mean   :3.597
 3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0   3rd Qu.:3.920
 Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0   Max.   :4.930
       wt             qsec             vs              am              gear
 Min.   :1.513   Min.   :14.50   Min.   :0.0000   Min.   :0.0000   Min.   :3.000
 1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:3.000
 Median :3.325   Median :17.71   Median :0.0000   Median :0.0000   Median
:4.000
```

```
 Mean   :3.217   Mean   :17.85   Mean   :0.4375   Mean   :0.4062   Mean   :3.688
 3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:4.000
 Max.   :5.424   Max.   :22.90   Max.   :1.0000   Max.   :1.0000   Max.   :5.000
     carb
 Min.   :1.000
 1st Qu.:2.000
 Median :2.000
 Mean   :2.812
 3rd Qu.:4.000
 Max.   :8.000

> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1

> #To select the following columns and (-) to hide the column
> s <- select(mtcars, hp, mpg, cyl, -am)
```

```
> head(s)
                   hp  mpg cyl
Mazda RX4         110 21.0   6
Mazda RX4 Wag     110 21.0   6
Datsun 710         93 22.8   4
Hornet 4 Drive    110 21.4   6
Hornet Sportabout 175 18.7   8
Valiant           105 18.1   6

> #To select columns with numeric indexes
> s2 <- select(mtcars,c(3:5))

> head(s2)
                  disp  hp drat
Mazda RX4          160 110 3.90
Mazda RX4 Wag      160 110 3.90
Datsun 710         108  93 3.85
Hornet 4 Drive     258 110 3.08
Hornet Sportabout  360 175 3.15
Valiant            225 105 2.76
```

```
> #To filter automatic and manual transmission cars 1 is for automatic and 0 for
manual

> f <- filter(mtcars, am == 1 )


> head(f)

                mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710     22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1


> #To filter cars with 6 or more cylinders and automatic transmission

> f2 <- filter(mtcars, cyl >= 6, am==1)


> head(f2)

                mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Ford Pantera L 15.8   8  351 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino   19.7   6  145 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora  15.0   8  301 335 3.54 3.570 14.60  0  1    5    8
```

```
> #filter with selected elements

> f3 <- filter(s, cyl == 8)


> print(f3)

                    hp  mpg cyl

Hornet Sportabout   175 18.7   8

Duster 360          245 14.3   8

Merc 450SE          180 16.4   8

Merc 450SL          180 17.3   8

Merc 450SLC         180 15.2   8

Cadillac Fleetwood  205 10.4   8

Lincoln Continental 215 10.4   8

Chrysler Imperial   230 14.7   8

Dodge Challenger    150 15.5   8

AMC Javelin         150 15.2   8

Camaro Z28          245 13.3   8

Pontiac Firebird    175 19.2   8

Ford Pantera L      264 15.8   8

Maserati Bora       335 15.0   8


> #To create a column "More Power" which stores TRUE if given condition is
TRUE
```

```
> newCol <- mutate(mtcars, MorePower = (hp >= 200))


> head(newCol)
   mpg cyl disp  hp drat    wt  qsec vs am gear carb MorePower
1 21.0   6  160 110 3.90 2.620 16.46  0  1    4    4     FALSE
2 21.0   6  160 110 3.90 2.875 17.02  0  1    4    4     FALSE
3 22.8   4  108  93 3.85 2.320 18.61  1  1    4    1     FALSE
4 21.4   6  258 110 3.08 3.215 19.44  1  0    3    1     FALSE
5 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2     FALSE
6 18.1   6  225 105 2.76 3.460 20.22  1  0    3    1     FALSE

> #To check how many cars satisfy condition
> table(newCol$MorePower)

FALSE  TRUE
   25     7

> #To arrange miles per gallon in ascending order
> arr <- arrange(newCol, mpg)

> head(arr)
   mpg cyl disp  hp drat    wt  qsec vs am gear carb MorePower
```

```
1 10.4   8  472 205 2.93 5.250 17.98  0  0    3    4     TRUE

2 10.4   8  460 215 3.00 5.424 17.82  0  0    3    4     TRUE

3 13.3   8  350 245 3.73 3.840 15.41  0  0    3    4     TRUE

4 14.3   8  360 245 3.21 3.570 15.84  0  0    3    4     TRUE

5 14.7   8  440 230 3.23 5.345 17.42  0  0    3    4     TRUE

6 15.0   8  301 335 3.54 3.570 14.60  0  1    5    8     TRUE
```

```
> #To arrange miles per gallon in descending order

> arr2 <- arrange(newCol, desc(mpg))
```

```
> head(arr2)

  mpg cyl  disp  hp drat    wt  qsec vs am gear carb MorePower

1 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1     FALSE

2 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1     FALSE

3 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2     FALSE

4 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2     FALSE

5 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1     FALSE

6 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2     FALSE
```

```
> #To arrange cars in ascending  order by miles per gallon

> head(mtcars[order(mtcars$mpg), ])

                  mpg cyl disp  hp drat    wt  qsec vs am gear carb
```

```
Cadillac Fleetwood  10.4   8  472 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8  460 215 3.00 5.424 17.82  0  0    3    4
Camaro Z28          13.3   8  350 245 3.73 3.840 15.41  0  0    3    4
Duster 360          14.3   8  360 245 3.21 3.570 15.84  0  0    3    4
Chrysler Imperial   14.7   8  440 230 3.23 5.345 17.42  0  0    3    4
Maserati Bora       15.0   8  301 335 3.54 3.570 14.60  0  1    5    8
```

```
> #To arrange cars in descending  order by miles per gallon
> head(mtcars[order(mtcars$mpg, decreasing = TRUE), ])
                  mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Lotus Europa   30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Fiat X1-9      27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
```

```
> #To arrange cars in order with cylinder, v shape and miles per gallon
> mtcars[with(mtcars, order(cyl, vs, mpg)), ]
                  mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8

```
Merc 450SLC        15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
AMC Javelin        15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Dodge Challenger   15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
Ford Pantera L     15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Merc 450SE         16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL         17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Hornet Sportabout  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Pontiac Firebird   19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2

> # summarize to find out (mean, median, mode, etc.)
> summarised <- summarise(arr, Mean.HorsePower = mean(hp))

> head(summarised)
  Mean.HorsePower
1        146.6875

> #To find mean horse power by cylinder, we use grouping as follows
> gp1 <- group_by(mtcars, cyl)

> grouped <- summarise(gp1, Mean.HorsePower = mean(hp))
`summarise()` ungrouping output (override with `.groups` argument)
```

```
> head(grouped)
# A tibble: 3 x 2
    cyl Mean.HorsePower
  <dbl>          <dbl>
1     4           82.6
2     6          122.
3     8          209.


> #Pipe operator
> #To get rows with the following conditions
> mtcars %>% filter(am == 1, hp > 200)
                mpg cyl disp  hp drat   wt qsec vs am gear carb
Ford Pantera L 15.8   8  351 264 4.22 3.17 14.5  0  1    5    4
Maserati Bora  15.0   8  301 335 3.54 3.57 14.6  0  1    5    8


> #To find mean miles per gallon by automatic transmission
> mtcars  %>% group_by(gear) %>% summarise(Mean.MPG = mean(mpg))
`summarise()` ungrouping output (override with `.groups` argument)
# A tibble: 3 x 2
   gear Mean.MPG
  <dbl>    <dbl>
1     3     16.1
```

```
2  4  24.5
3  5  21.4
>
```