Statistics Practical

1. Use the internal/own database and run the following operators also explain the output.
data(), dim(), names(), View(), str(), ls(), rm()

#data() returns a list of currently loaded datasets or loads a dataset.

#dim() is used to get or set the dimension of the specified matrix, array or data frame.

#name() returns names of the columns

#The View() invokes a spreadsheet-style data viewer on a matrix-like R object.

#str() is used for compactly displaying the internal structure of a R object.

#ls() is used to list the names of all the objects that are present in the working directory.

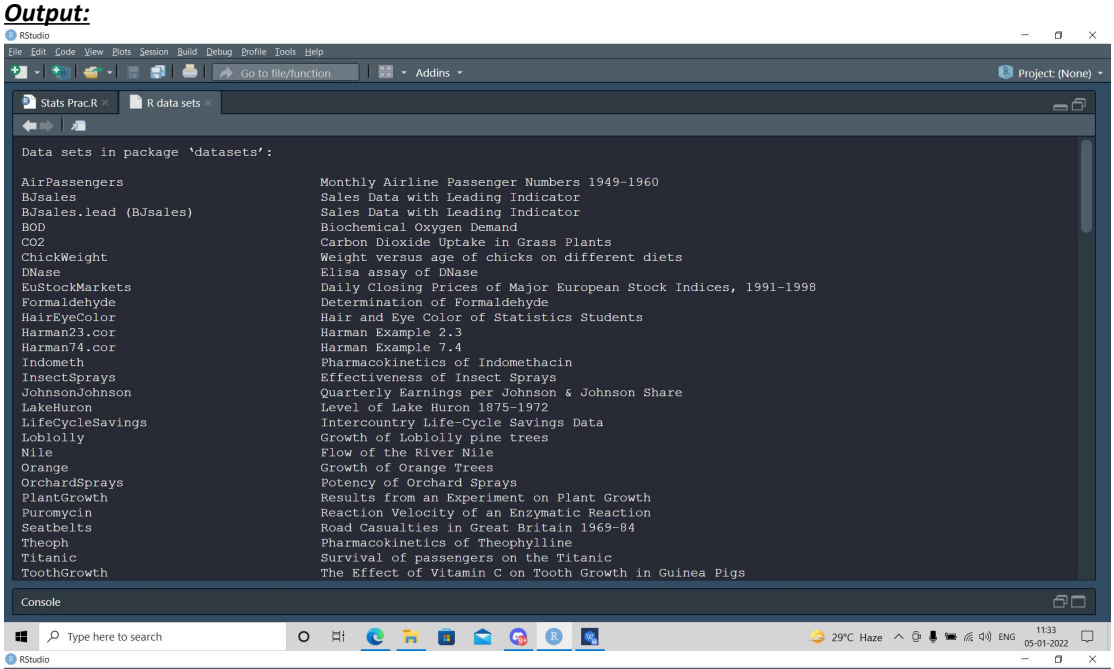
#lists all objects from the memory

#rm() is used to delete objects from the memory.

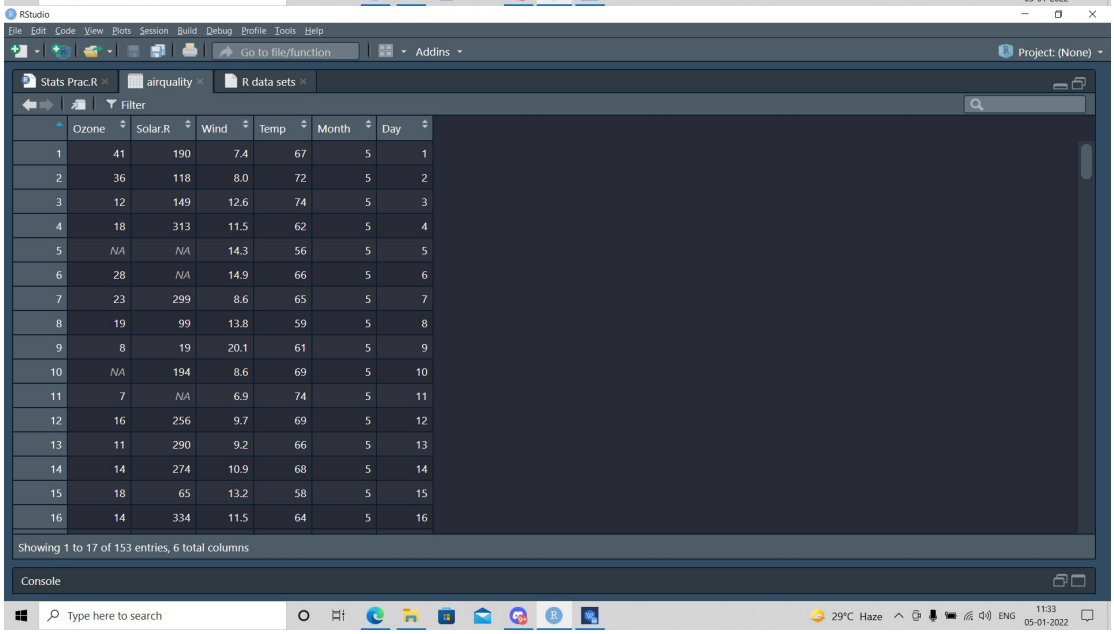
```r
#Jay Rathod
#Statistics Practical

#data() returns a list of currently loaded datasets or loads a dataset.
data()

#Prints first 10 rows of the dataset
head(airquality, 10)

#dim() is used to get or set the dimension of the specified matrix, array or data frame.
dim(airquality)

#name() returns names of the columns
names(airquality)

#The View() invokes a spreadsheet-style data viewer on a matrix-like R object.
View(airquality)

#str() is used for compactly displaying the internal structure of a R object.
str(airquality)

#ls() is used to list the names of all the objects that are present in the working directory.
ls(airquality)
#lists all objects from the memory
ls()

#rm() is used to delete objects from the memory.
rm("actuals_preds")
```

```r
#data() returns a list of currently loaded datasets or loads a dataset.
data()

#Prints first 10 rows of the dataset
head(airquality, 10)

#dim() is used to get or set the dimension of the specified matrix, array or data frame.
dim(airquality)

#name() returns names of the columns
names(airquality)

#The View() invokes a spreadsheet-style data viewer on a matrix-like R object.
View(airquality)

#str() is used for compactly displaying the internal structure of a R object.
str(airquality)

#ls() is used to list the names of all the objects that are present in the working directory.
ls(airquality)
#lists all objects from the memory
ls()

#rm() is used to delete objects from the memory.
rm("actuals_preds")
# It can be used with ls() function to delete all objects.
rm(list=ls())
```

Output:



```
Data sets in package 'datasets':

AirPassengers                       Monthly Airline Passenger Numbers 1949-1960
BJsales                             Sales Data with Leading Indicator
BJsales.lead (BJsales)              Sales Data with Leading Indicator
BOD                                 Biochemical Oxygen Demand
CO2                                 Carbon Dioxide Uptake in Grass Plants
ChickWeight                         Weight versus age of chicks on different diets
DNase                               Elisa assay of DNase
EuStockMarkets                      Daily Closing Prices of Major European Stock Indices, 1991-1998
Formaldehyde                        Determination of Formaldehyde
HairEyeColor                        Hair and Eye Color of Statistics Students
Harman23.cor                        Harman Example 2.3
Harman74.cor                        Harman Example 7.4
Indometh                            Pharmacokinetics of Indomethacin
InsectSprays                        Effectiveness of Insect Sprays
JohnsonJohnson                      Quarterly Earnings per Johnson & Johnson Share
LakeHuron                           Level of Lake Huron 1875-1972
LifeCycleSavings                    Intercountry Life-Cycle Savings Data
Loblolly                            Growth of Loblolly pine trees
Nile                                Flow of the River Nile
Orange                              Growth of Orange Trees
OrchardSprays                       Potency of Orchard Sprays
PlantGrowth                         Results from an Experiment on Plant Growth
Puromycin                           Reaction Velocity of an Enzymatic Reaction
Seatbelts                           Road Casualties in Great Britain 1969-84
Theoph                              Pharmacokinetics of Theophylline
Titanic                             Survival of passengers on the Titanic
ToothGrowth                         The Effect of Vitamin C on Tooth Growth in Guinea Pigs
```



	Ozone	Solar.R	Wind	Temp	Month	Day
1	41	190	7.4	67	5	1
2	36	118	8.0	72	5	2
3	12	149	12.6	74	5	3
4	18	313	11.5	62	5	4
5	NA	NA	14.3	56	5	5
6	28	NA	14.9	66	5	6
7	23	299	8.6	65	5	7
8	19	99	13.8	59	5	8
9	8	19	20.1	61	5	9
10	NA	194	8.6	69	5	10
11	7	NA	6.9	74	5	11
12	16	256	9.7	69	5	12
13	11	290	9.2	66	5	13
14	14	274	10.9	68	5	14
15	18	65	13.2	58	5	15
16	14	334	11.5	64	5	16

Showing 1 to 17 of 153 entries, 6 total columns

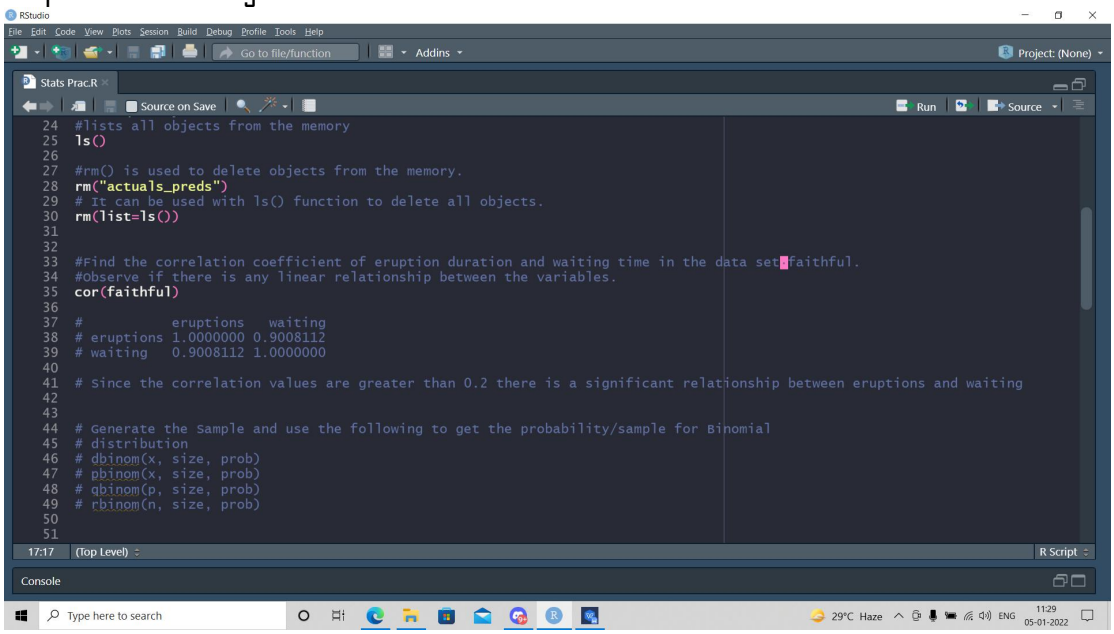
```
> #data() returns a list of currently loaded datasets or loads a dataset.
> data()
> #Prints first 10 rows of the dataset
> head(airquality, 10)
   Ozone Solar.R Wind Temp Month Day
1     41     190  7.4   67     5   1
2     36     118  8.0   72     5   2
3     12     149 12.6   74     5   3
4     18     313 11.5   62     5   4
5     NA      NA 14.3   56     5   5
6     28      NA 14.9   66     5   6
7     23     299  8.6   65     5   7
8     19      99 13.8   59     5   8
9      8      19 20.1   61     5   9
10    NA     194  8.6   69     5  10
> #dim() is used to get or set the dimension of the specified matrix, array or data frame.
> dim(airquality)
[1] 153   6
> #name() returns names of the columns
> names(airquality)
[1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
> #The View() invokes a spreadsheet-style data viewer on a matrix-like R object.
> View(airquality)
> #str() is used for compactly displaying the internal structure of a R object.
> str(airquality)
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
```

```
[1] 153   6
> #name() returns names of the columns
> names(airquality)
[1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
> #The View() invokes a spreadsheet-style data viewer on a matrix-like R object.
> View(airquality)
> #str() is used for compactly displaying the internal structure of a R object.
> str(airquality)
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
> #ls() is used to list the names of all the objects that are present in the working directory.
> ls(airquality)
[1] "Day"     "Month"   "Ozone"   "Solar.R" "Temp"    "Wind"
> #lists all objects from the memory
> ls()
[1] "coin"  "flips" "freqs" "p"     "s"
> #rm() is used to delete objects from the memory.
> rm("actuals_preds")
Warning message:
In rm("actuals_preds") : object 'actuals_preds' not found
> # It can be used with ls() function to delete all objects.
> rm(list=ls())
>
```

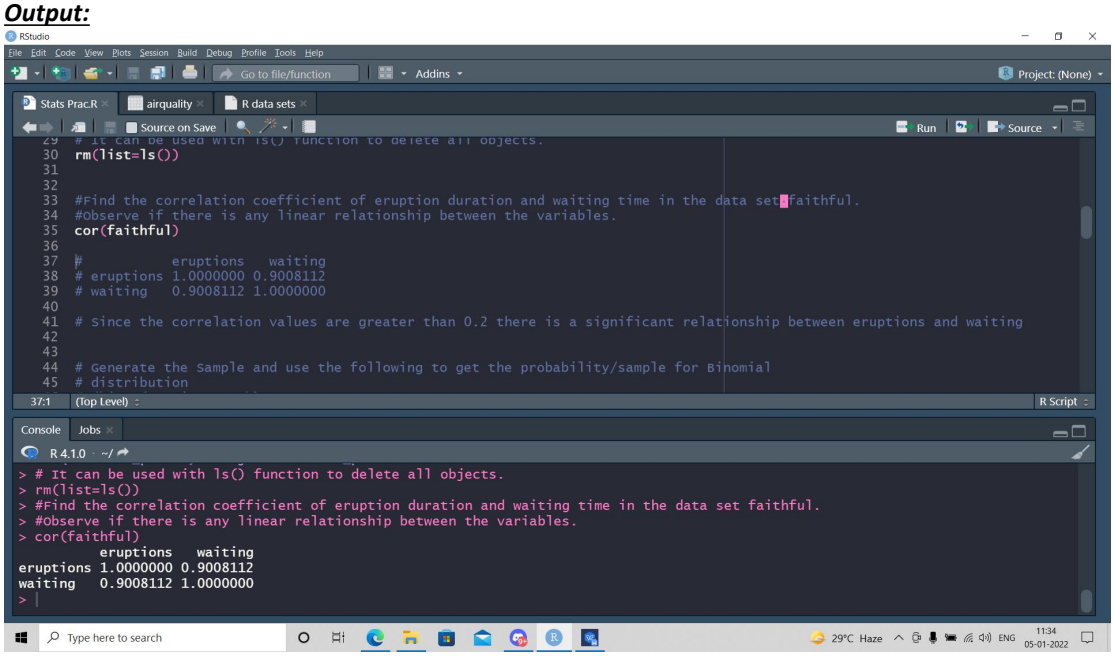
2. Find the correlation coefficient of eruption duration and waiting time in the data set faithful.
Observe if there is any linear relationship between the variables.

```
#           eruptions   waiting
# eruptions 1.0000000 0.9008112
# waiting   0.9008112 1.0000000
```

Since the correlation values are greater than 0.2 there is a significant relationship between eruptions and waiting



Output:



3. Generate the Sample and use the following to get the probability/sample for Binomial distribution

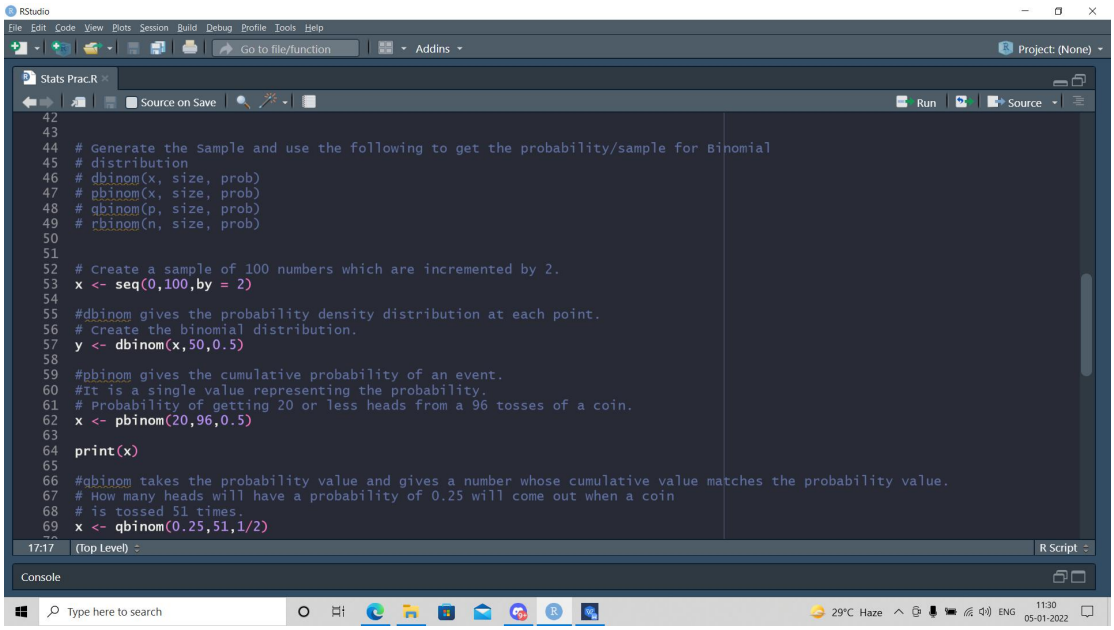
dbinom(x, size, prob)
pbinom(x, size, prob)
qbinom(p, size, prob)
rbinom(n, size, prob)

#dbinom gives the probability density distribution at each point.

#pbinom gives the cumulative probability of an event.

#qbinom takes the probability value and gives a number whose cumulative value matches the probability value.

#rbimon generates required number of random values of given probability from a given sample



```r
print(x)

#qbinom takes the probability value and gives a number whose cumulative value matches the probability value.
# How many heads will have a probability of 0.25 will come out when a coin
# is tossed 51 times.
x <- qbinom(0.25,51,1/2)

print(x)

#rbimon generates required number of random values of given probability from a given sample
# Find 8 random values from a sample of 150 with probability of 0.4.
x <- rbinom(8,150,0.4)

print(x)


# Generate the random sample between 4 to 8 where

# i. replacement is allowed
#means numbers can be repeated

sample(4:8,10, replace=TRUE)


# ii. replacement is not allowed
#means numbers cannot be repeated
sample(4:8,5, replace=FALSE)
```

Output:

```
> #Observe if there is any linear relationship between the variables.
> cor(faithful)
          eruptions   waiting
eruptions 1.0000000 0.9008112
waiting   0.9008112 1.0000000
> # Create a sample of 100 numbers which are incremented by 2.
> x <- seq(0,100,by = 2)
> #dbinom gives the probability density distribution at each point.
> # Create the binomial distribution.
> y <- dbinom(x,50,0.5)
> #pbinom gives the cumulative probability of an event.
> #It is a single value representing the probability.
> # Probability of getting 20 or less heads from a 96 tosses of a coin.
> x <- pbinom(20,96,0.5)
> print(x)
[1] 3.659684e-09
> #qbinom takes the probability value and gives a number whose cumulative value matches the probability value.
> # How many heads will have a probability of 0.25 will come out when a coin
> # is tossed 51 times.
> x <- qbinom(0.25,51,1/2)
> print(x)
[1] 23
> #rbimon generates required number of random values of given probability from a given sample
> # Find 8 random values from a sample of 150 with probability of 0.4.
> x <- rbinom(8,150,0.4)
> print(x)
[1] 56 61 68 55 71 64 54 59
>
```

4. Generate the random sample between 4 to 8 where
i. replacement is allowed
ii. replacement is not allowed

i. replacement is allowed
#means numbers can be repeated

ii. replacement is not allowed
#means numbers cannot be repeated

```
#rbinom generates required number of random values of given probability from a given sample
# Find 8 random values from a sample of 150 with probability of 0.4.
x <- rbinom(8,150,0.4)

print(x)


# Generate the random sample between 4 to 8 where

# i. replacement is allowed
#means numbers can be repeated

sample(4:8,10, replace=TRUE)


# ii. replacement is not allowed
#means numbers cannot be repeated
sample(4:8,5, replace=FALSE)


# If the coin is tossed 3 times generate the sample in tabular form along with the probability of each combination.
library(prob)

s <- tosscoin(3)

print(s)
```

Output:

```
#means numbers can be repeated

sample(4:8,10, replace=TRUE)


# ii. replacement is not allowed
#means numbers cannot be repeated
sample(4:8,5, replace=FALSE)


# If the coin is tossed 3 times generate the sample in tabular form along with the probability of each combination.
library(prob)

s <- tosscoin(3)
```

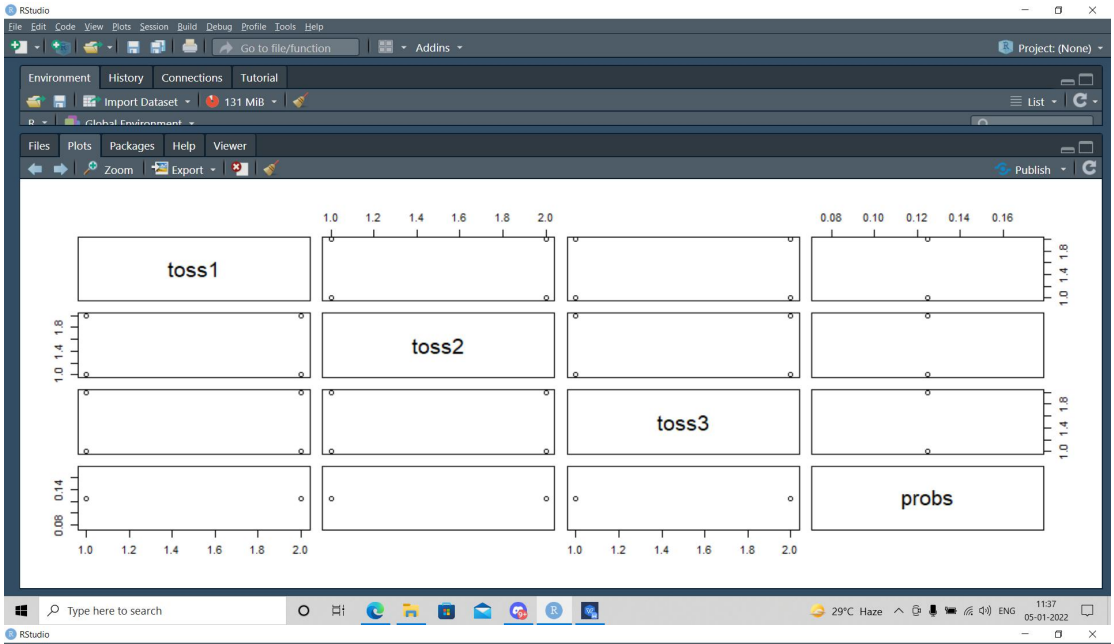
```
> # Find 8 random values from a sample of 150 with probability of 0.4.
> x <- rbinom(8,150,0.4)
> print(x)
[1] 56 61 68 55 71 64 54 59
> sample(4:8,10, replace=TRUE)
 [1] 6 8 4 8 7 5 5 6 7 4
> # ii. replacement is not allowed
> #means numbers cannot be repeated
> sample(4:8,5, replace=FALSE)
[1] 8 4 5 6 7
>
```

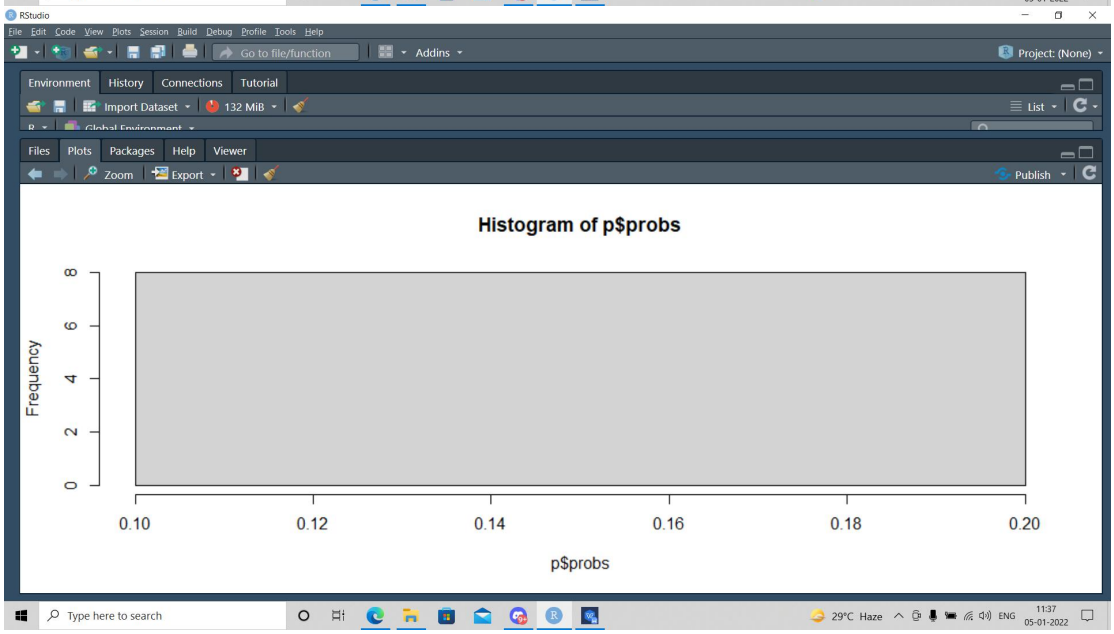
5. If the coin is tossed 3 times generate the sample in tabular form along with the probability of each combination. Plot the Histogram and scatterplot using R internal data set.

```r
sample(4:8,10, replace=TRUE)


# ii. replacement is not allowed
#means numbers cannot be repeated
sample(4:8,5, replace=FALSE)


# If the coin is tossed 3 times generate the sample in tabular form along with the probability of each combination.
library(prob)

s <- tosscoin(3)

print(s)

#probablity of each outcome
p <- probspace(tosscoin(3))


#Plot the Histogram and scatterplot using R internal data set.

#scatter plot
plot(p)

#histogram plot
hist(p$probs, )
```

Output:

```
[1] 56 61 68 55 71 64 54 59
> sample(4:8,10, replace=TRUE)
 [1] 6 8 4 8 7 5 5 6 7 4
> # ii. replacement is not allowed
> #means numbers cannot be repeated
> sample(4:8,5, replace=FALSE)
[1] 8 4 5 6 7
> # If the coin is tossed 3 times generate the sample in tabular form along with the probability of each combination.
> library(prob)
> s <- tosscoin(3)
> print(s)
  toss1 toss2 toss3
1     H     H     H
2     T     H     H
3     H     T     H
4     T     T     H
5     H     H     T
6     T     H     T
7     H     T     T
8     T     T     T
> #probablity of each outcome
> p <- probspace(tosscoin(3))
> #scatter plot
> plot(p)
> #histogram plot
> hist(p$probs, )
>
>
```


toss1
toss2
toss3
probs


Histogram of p$probs
Frequency
p$probs