1.Addition:

```
num1=as.integer(readline(prompt = "Enter the num 1:"))

num2 =as.integer(readline(prompt = "Enter a number2:"))

sum=num1+num2

print((paste("sum:",sum)))
```

2.Mean:

```
height <-c(150,174, 138, 186, 128, 136, 171, 163, 152, 131)

result.mean <-mean(height)

print(result.mean)
```

3.Bar plot:

```
temperatures <- c(20, 22, 25, 29, 23, 27, 28)

result <- barplot(temperatures,

        main = "Maximum Temperatures in a Week",

        xlab = "Degree Celsius",

        ylab = "Day",

        names.arg = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"),

        col = "blue",

)

print(result)
```

4.Box plot:

```
b <- c(10,12,13,14,17,19,20,30,50,70,90,100)
print(boxplot(b,col="green"))
```

5.Decision tree:

```
library(rpart)
library(rpart.plot)
data=read.csv("C:\\Users\\arunk\\OneDrive\\Desktop\\DWDM\\Gender.csv")
tree <- rpart(Height ~ Gender+Weight,data)
a <- data.frame(Gender=c("Male"),Weight=c(85))
result <- predict(tree,a)
print(result)
rpart.plot(tree)
tree1 <- rpart(Gender~ Height+Weight,data)
a <- data.frame(Height=c(170),Weight=c(85))
result <- predict(tree,a)
print(result)
rpart.plot(tree1)
```

6.Division:

```
num1=as.integer(readline(prompt = "Enter the number 1:"))

num2 =as.integer(readline(prompt = "Enter a number2:"))

div=num1/num2

print((paste("Division:",div)))
```

7.Histogram:

```
temperatures <- c(20, 22, 25, 29, 23, 27, 28)


result <- hist(temperatures,

               main = "Maximum Temperatures in a Week",

               xlab = "Degree Celsius",

               ylab = "Day",

               names.arg = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"),

               col="green"

)

print(result)
```

8.Linear regression:

```
x <-c(150,174, 138, 186, 128, 136, 171, 163, 152, 131)

y<-c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

relation <-lm(y~x)

print(summary(relation))

a <-data.frame(x=170)

result <- predict(relation,a)

print(result)

png(file = "linear_regression.png")

plot(y,x,col = "red",main = "Height and Weight Regression",abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab = "Height in cm")

dev.off()
```

9.Median:

```
height <-c(150,174, 138, 186, 128, 136, 171, 163, 152, 131)

result.median <-median(height)

print(result.median)
```

10.Min max normalization:

```
original_vector <- c(10, 20, 30, 40, 50)

normalized_vector<-(original_vector- min(original_vector)) / (max(original_vector) - min(original_vector))

print(normalized_vector)


original_vector <- c(100, 200, 309, 40, 50,60,70,80,90,10)

normalized_vector<-(original_vector- min(original_vector)) / (max(original_vector) - min(original_vector))

print(normalized_vector)
```

11.Mode:

```
getmode <- function(v)
  {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
v <- c(150,174, 138, 186, 128, 136, 171, 163, 152, 131,171,131,171)
result <- getmode(v)
print(result)
```

12.Multiple Regression:

```
d=read.csv("C:\\Users\\arunk\\OneDrive\\Desktop\\DWDM\\set1.csv")
View(d)
summary(d)
plot(d$Glucose,d$DiabetesPedigreeFunction)
p1=runif(nrow(d))
p2=order(p1)
training_ds=d[p2[1:25],]
test_ds=d[p2[26:39],]
Multiple_resgression=lm(DiabetesPedigreeFunction~Glucose+Age,
data=training_ds)
abline(Multiple_resgression,col="red")
summary(Multiple_resgression)
plot(Multiple_resgression)

pred_values=predict(Multiple_resgression,newdata = test_ds)
test_ds$pred_DiabetesPedigreeFunction=pred_values
View(test_ds)
```

13.Multiplication:

```
num1=as.integer(readline(prompt = "Enter the num 1:"))
num2 =as.integer(readline(prompt = "Enter a number2:"))
mul=num1*num2
print((paste("Multiplication:",mul)))
```

14.odd or Even:

```
num =as.integer(readline(prompt = "Enter a number:"))
if (num %% 2 ==0){
  print(paste(num,"is Even number!!"))
}else{
  print(paste(num,"is Odd number!!"))
}
```

15.pie Chart:

```
a <- c(80,70,50,60,70,100)
result<-
(pie(a,main="piechart",labels=c("student1","student2","student3","student
4","student 5","student 6"),
                  col = c("red", "orange", "yellow", "blue", "green","black")))
print(result)
```

16.Quantile:

```
names<-c("Ram","Shyam","Kumar")
age<-c(23,24,35)
marks<-c(88,78,25)
df<-data.frame(names,age,marks)
quantile(df $age)
write.csv(df,"datafr.csv")
```

17.Range:

```
names<-c("Ram","Shyam","Kumar")

age<-c(23,24,35)

marks<-c(88,78,25)

df<-data.frame(names,age,marks)

range(df $age)

write.csv(df,"datafr.csv")
```

18.Scatter plot:

```
input <- mtcars[,c('wt','mpg')]

print(head(input))

plot(x = input$wt, y = input$mpg,

    xlab = "Weight",

    ylab = "Milage",

    xlim = c(0.5, 3.5),

    ylim = c(15, 30),

    main = "Weight vs Milage"

)
```

19.Subtraction:

```
num1=as.integer(readline(prompt = "Enter the num 1:"))

num2 =as.integer(readline(prompt = "Enter a number2:"))

sub=num1-num2

print((paste("subraction value:",sub)))
```

20.Z-Score normalization:

```
original_vector <- c(3,5,5,8,9,12,12,13,15,16,17,19,22,24,25,134)

x <-mean(original_vector)

print(paste("Mean:",x))

u <-sd(original_vector)

print(paste("S.D:",u))

normalized_vector <- (original_vector - x) / u

print(normalized vector)
```

21.K-Means:

```
# Load a dataset

data(iris)

# Select the variables to be used for clustering

x <- iris[, c("Sepal. Length", "Sepal. Width", "Petal. Length", "Petal. Width")]

# Perform K-means clustering with K=3

kmeans_model <- kmeans(x, centers = 3)

# Print the results

kmeans_model

# Create a scatterplot of the first two variables with points colored by cluster

library(ggplot2)

ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width,color=factor(kmeans_model$cluste))
) +geom_point()
```

22.Normal Distribution:

```
x <- rnorm(100, mean = 0, sd = 1)

hist(x)

dnorm(1, mean = 0, sd = 1)

pnorm(1, mean = 0, sd = 1)
```

23.Array:

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

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

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

print(result)
```

24.Square Root:

```
x <- 4

sqrt(x)
```

25.Line Chart:

```
v <- c (17, 25, 38, 13, 41)

plot (v, type = "o")
```

24.Random Forest:

```
install.packages("caTools")
install.packages("randomForest")
library(caTools)
library(randomForest)
split <- sample.split(iris, SplitRatio = 0.7)
split

train <- subset(iris, split == "TRUE")
test <- subset(iris, split == "FALSE")
set.seed(120)
classifier_RF = randomForest(x = train[-5],y = train$Species,ntree = 500)
classifier_RF
y_pred = predict(classifier_RF, newdata = test[-5])
confusion_mtx = table(test[, 5], y_pred)
confusion_mtx
plot(classifier_RF)
importance(classifier_RF)
varImpPlot(classifier_RF)
```

26.Confusion Matrix:

```
set.seed(123)

data <- data.frame(Actual = sample(c("True","False"), 100, replace = TRUE),

                Prediction = sample(c("True","False"), 100, replace = TRUE)

)

table (data$Prediction, data$Actual)
```

27.Chi Square:

```
library (MASS)

print(str(survey))

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

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

print(stu_data)

print(chisq.test(stu_data))
```

29.Decimal Scaling:

```
library(caret)

gfg <- c(244,753,596,645,874,141,639,465,999,654)

ss <- preProcess(as.data.frame(gfg), method=c("range"))

gfg <- predict(ss, as.data.frame(gfg))

gfg
```

30.Apriori Algorithm:

```
library(arules)
library(arulesViz)

library(RColorBrewer)

data("Groceries")

rules <- apriori(Groceries, parameter = list(supp = 0.01, conf = 0.2))

inspect(rules[1:10])

arules::itemFrequencyPlot(Groceries, topN = 20,col = brewer.pal(8, 'Pastel2'), main
= 'Relative Item Frequency Plot',type = "relative",ylab = "Item Frequency (Relative)")
```