a. What will the program do?

-The program will take the grocery shopping list and perform some tasks on it. We will have to clean the data to make it more readable and easily accessible.

b. What the input to the program will be.

- **datasetPath: User should input the full path of the csv file**
- **numberOfClusters: Number between 2 and 4**
- **minSupport : Number between 0.001 and 1**
- **minConfidence :Number between 0.001 and 1**

c. What the output from the program will be.

Will be discussed later

First:we will import the dataset by using =(library called “library(readr)”

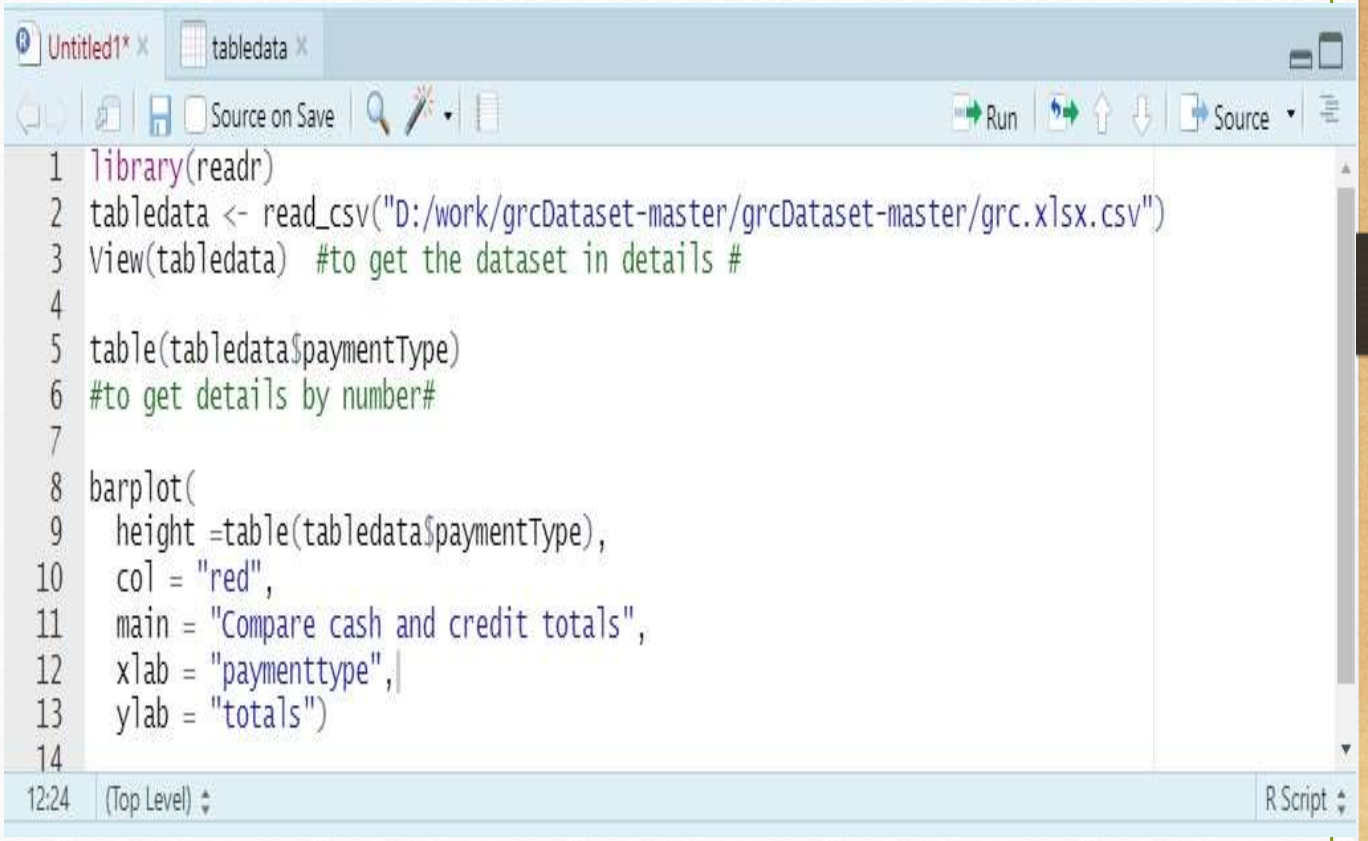
---{The code }---

```
library(readr)
datasetPath<- readline("Enter path of csv file: ")
grc<read_csv(datasetPath)
```

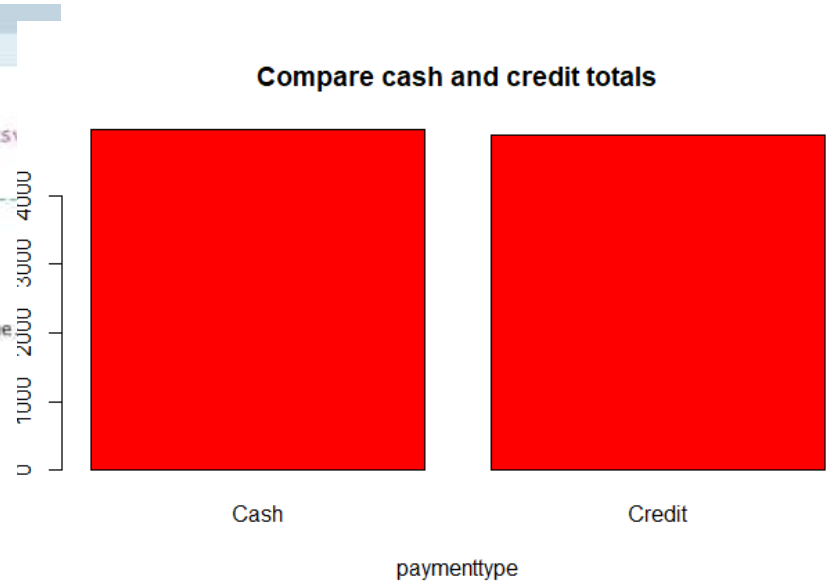
b) We will use a different type of Data Visualization tools:

i) Compare cash and credit totals.

```
library(readr)
tabledata <- read_csv("D:/work/grcDataset-master/grcDataset-master/grc.xlsx.csv")
View(tabledata)  #to get the dataset in details #

table(tabledata$paymentType)
 #to get details by number#

barplot(
  height =table(tabledata$paymentType),
  col = "red",
  main = "Compare cash and credit totals",
  xlab = "paymenttype",
  ylab = "totals")
```



```
Console   Terminal   Jobs
R 4.1.1 · ~/
> library(readr)
Warning message:
package 'readr' was built under R version 4.1.2
> tabledata <- read_csv("D:/work/grcDataset-master/grcDataset-master/grc.xlsx.cs
Rows: 9835 Columns: 8

-- Column specification --------------------------------------------------------
Delimiter: ","
chr (4): items, customer, city, paymentType
dbl (4): count, total, rnd, age

i Use spec() to retrieve the full column specification for this data.
i Specify the column types or set show_col_types = FALSE to quiet this message.
> View(tabledata)  #to get the dataset in details #
>
> table(tabledata$paymentType)

  Cash Credit
  4957   4878
> #to get details by number#
>
> barplot(
+   height =table(tabledata$paymentType),
+   col = "red",
+   main = "Compare cash and credit totals",
+   xlab = "paymenttype",
+   ylab = "totals")
```

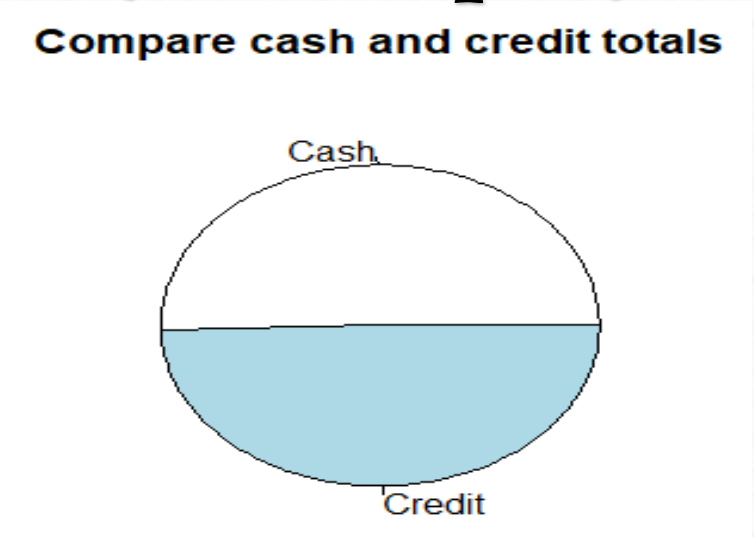


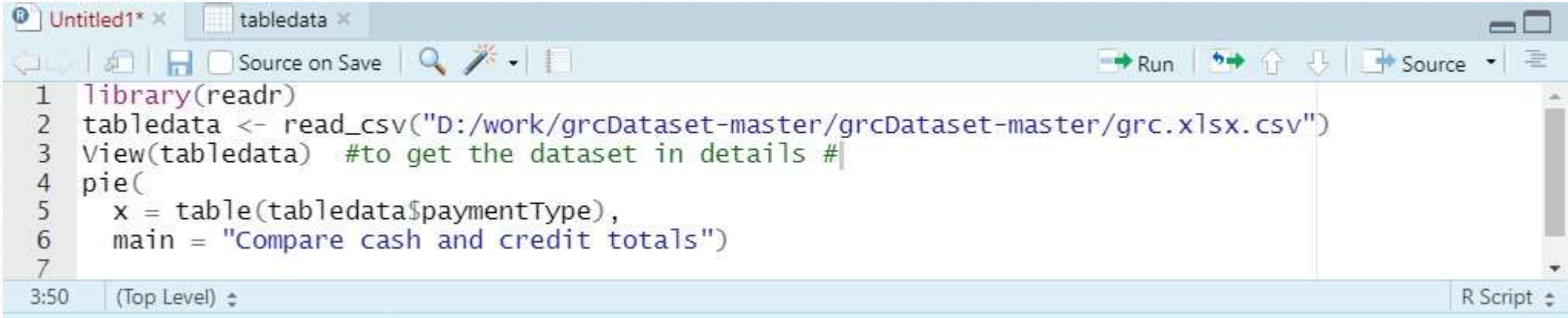
---{The output}---

Another way of getting cash and credit totals

```
library(readr)
tabledata <- read_csv("D:/work/grcDataset-master/grcDataset-master/grc.xlsx.csv")
View(tabledata)

#by using pie #
pie(
  x = table(tabledata$paymentType),
  main = "Compare cash and credit totals")
```

{the output}

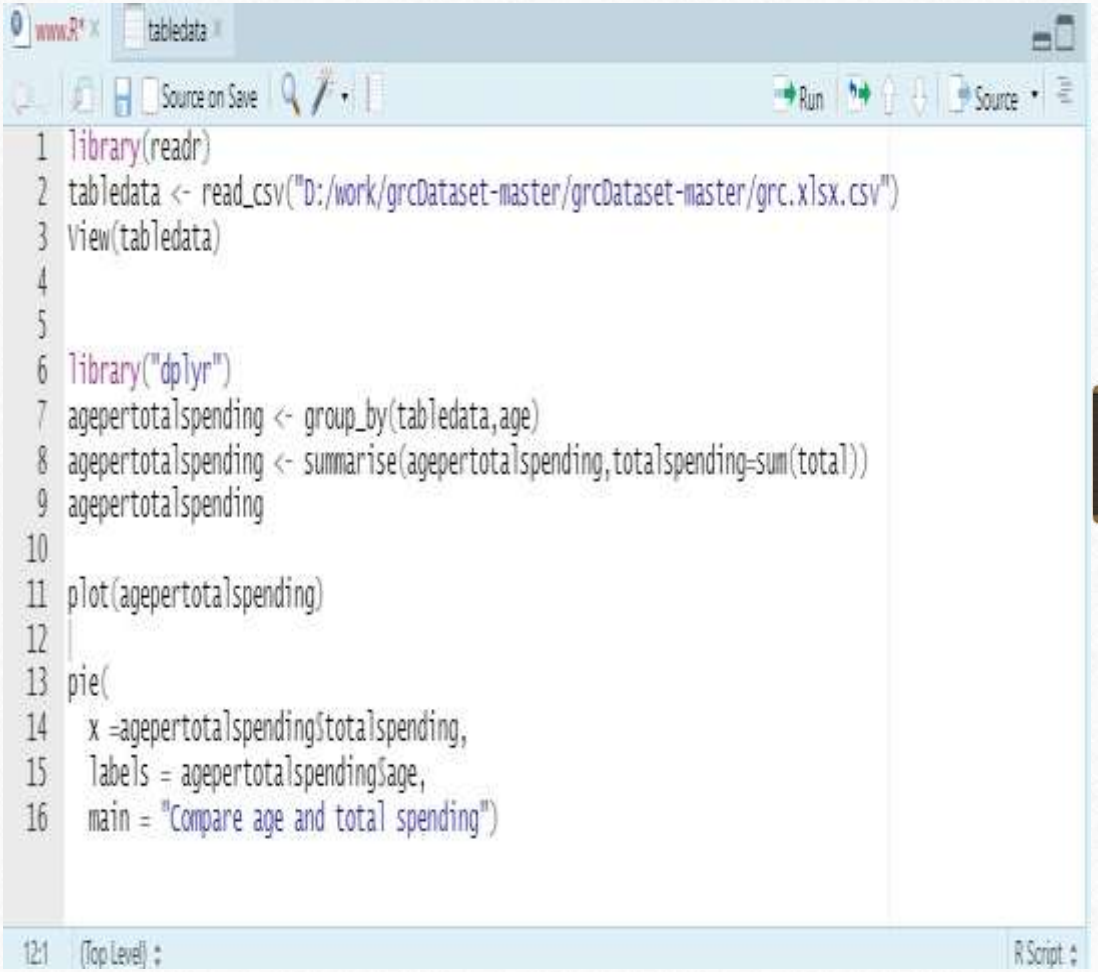


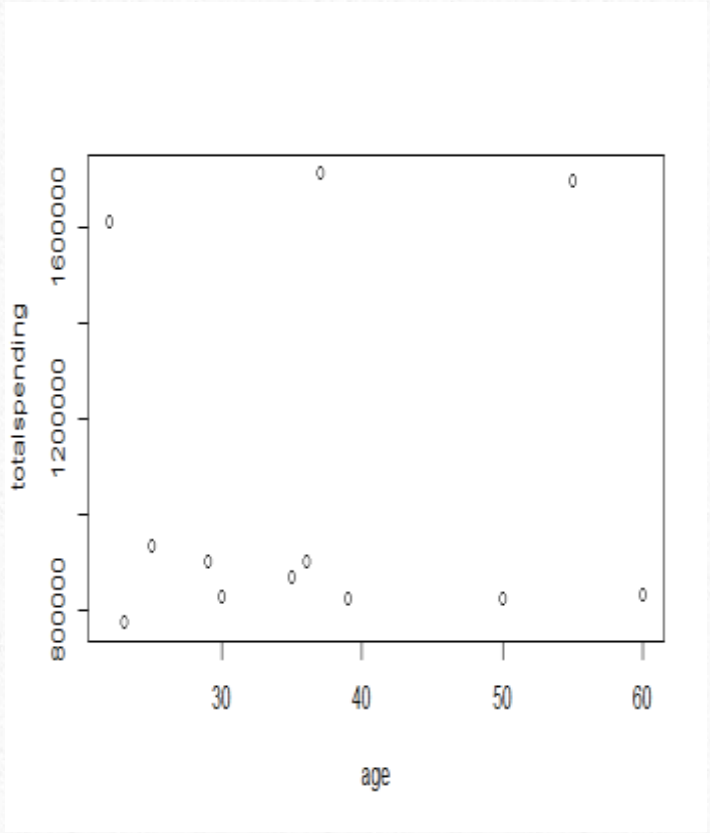


ii. Compare each age and sum of total spending:

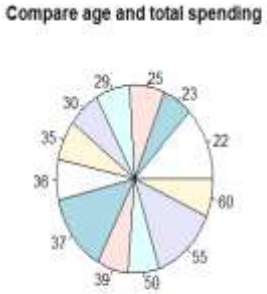
We use ==(“dplyr”)==library

```
library("dplyr")
agepertotalspending <- group_by(tabledata,age)
agepertotalspending <-
summarise(agepertotalspending,totalspending=sum(total))
agepertotalspending
                          #to show numric details #
plot(agepertotalspending)        #to scaterplot it#

pie(
  x =agepertotalspending$totalspending,
  labels = agepertotalspending$age,        # to piechar it#
  main = "Compare age and total spending")
```





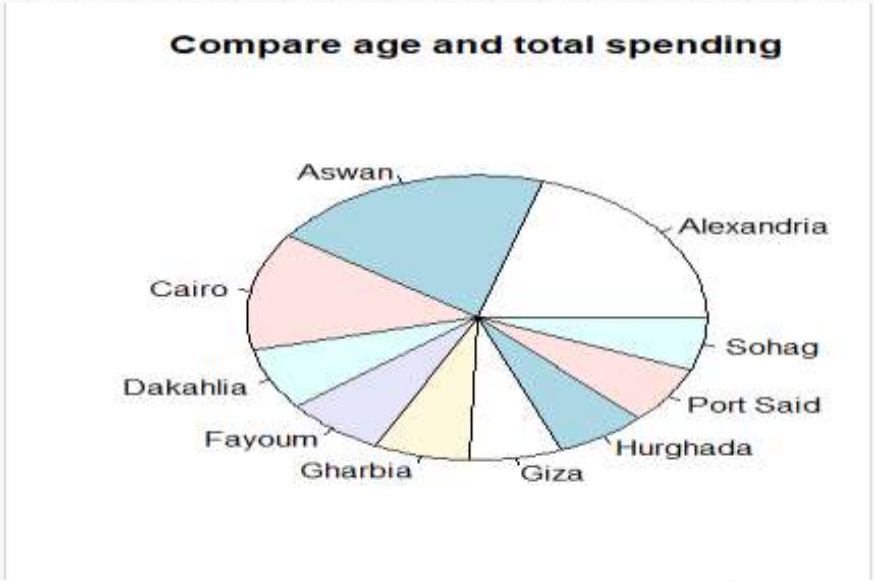

```
Console  Terminal  Jobs
R 4.1.1 · ~/
> library("dplyr")
> agepertotalspending <- group_by(tabledata,age)
> agepertotalspending <- summarise(agepertotalspending,totalspending=sum(total))
> agepertotalspending
# A tibble: 12 x 2
     age totalspending
   <dbl>         <dbl>
 1    22       1613801
 2    23        772871
 3    25        932250
 4    29        900797
 5    30        829587
 6    35        869668
 7    36        901010
 8    37       1714689
 9    39        825147
10    50        824064
11    55       1699068
12    60        831272
>
> plot(agepertotalspending)
>
> pie(
+   x =agepertotalspending$totalspending,
+   labels = agepertotalspending$age,
+   main = "Compare age and total spending")
>
```



---{The output}---

iii. Show each city total spending and arrange it by total descending. ---{The output}---

```
library("dplyr")
cityperspending<- group_by(tabledata,city)
cityperspending <-
summarise(cityperspending,totalspending=sum(total))
cityperspending

cityperspending
<- sort(cityperspending$totalspending, decreasing=TRUE)
```



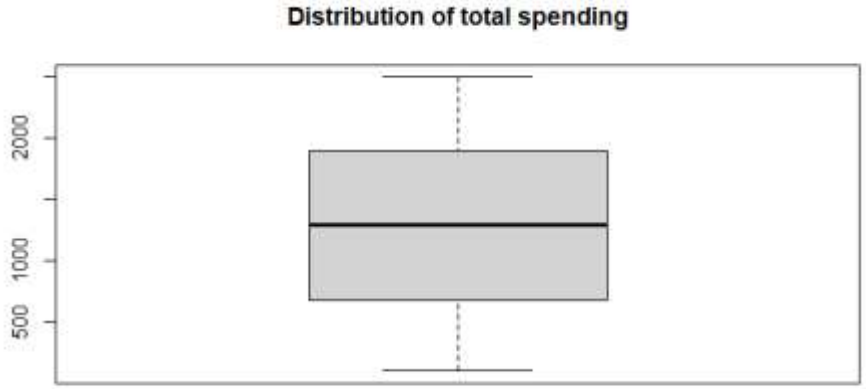

```
pie(  x = cityperspending$totaspending,
      labels = cityperspending$city,
      main = "Compare age and total spending")
```

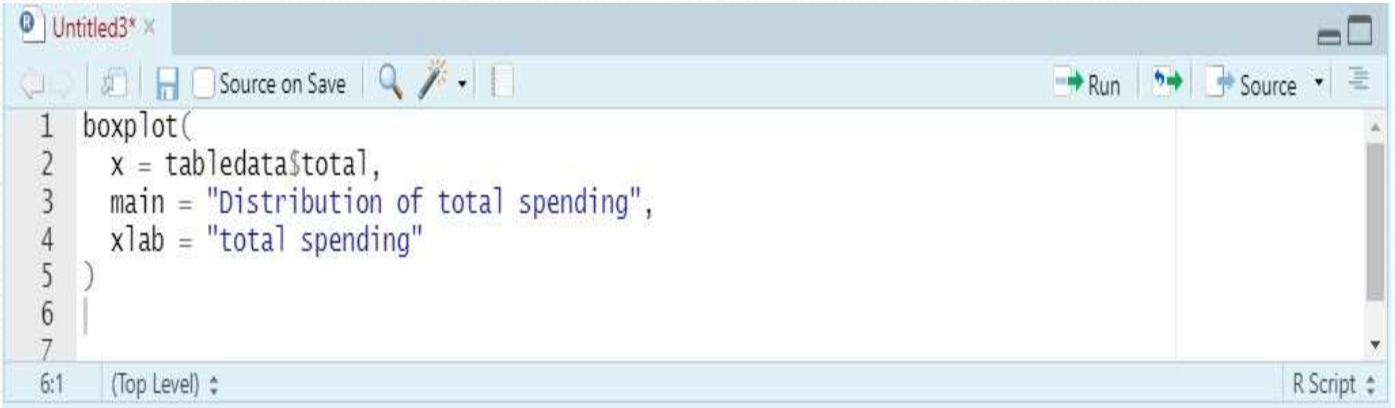
```
255
256
257
258 library("dplyr")
259 cityperspending<- group_by(grc,city)
260 cityperspending <- summarise(cityperspending,totalspending=sum(total))
261 cityperspending
262
263
264
265
266 citiesSpending<- sort(cityperspending$totalspending, decreasing=TRUE)
267 pie(  x =citiesSpending,
268       labels = cityperspending$city,
269       main = "Compare age and total spending")
270
271
272
273
```

iv. Display the distribution of total spending.

--{The out put}--

```
boxplot(
  x = tabledata$total,
  main = "Distribution of total
spending",
  xlab = "total spending"
)
```

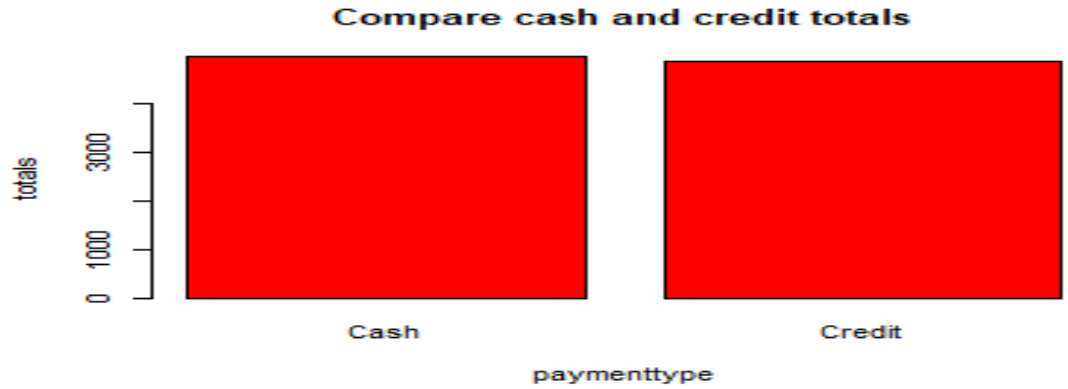


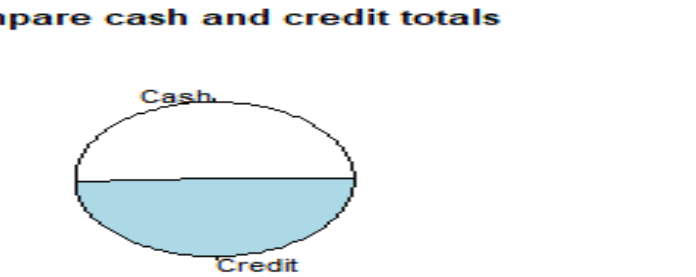


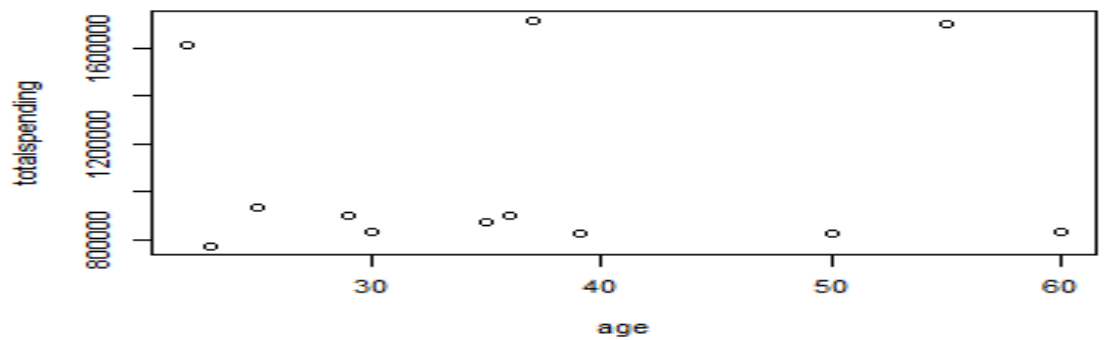
c)Put all previous plots in one dashboard.

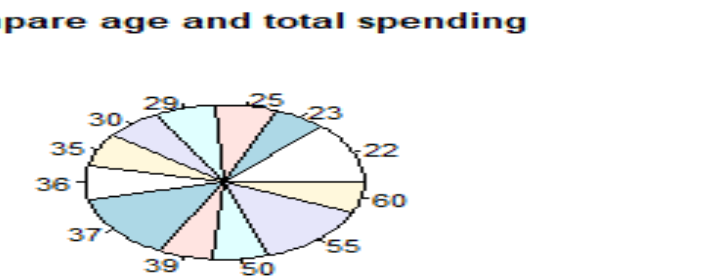
```
library(readr)
tabledata <- read_csv("D:/work/grcDataset-
master/grcDataset-master/grc.xlsx.csv")
View(tabledata)


par(mfrow=c(3,2))
barplot(
  height =table(tabledata$paymentType),
  col = "red",
  main = "Compare cash and credit totals",
  xlab = "paymenttype",
  ylab = "totals")

pie(
  x = table(tabledata$paymentType),
  main = "Compare cash and credit totals")
```

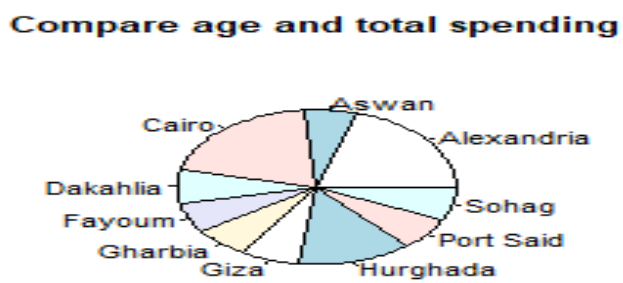
```
plot(agepertotalspending)        #to scaterplot it#

pie(
  x =agepertotalspending$totalspending,
  labels = agepertotalspending$age,       # to piechar it#
  main = "Compare age and total spending")

pie(  x = cityperspending$totalspending,
      labels = cityperspending$city,
      main = "Compare age and total spending")


boxplot(
  x = tabledata$total,
  main = "Distribution of total
spending",
  xlab = "total spending"
)
```

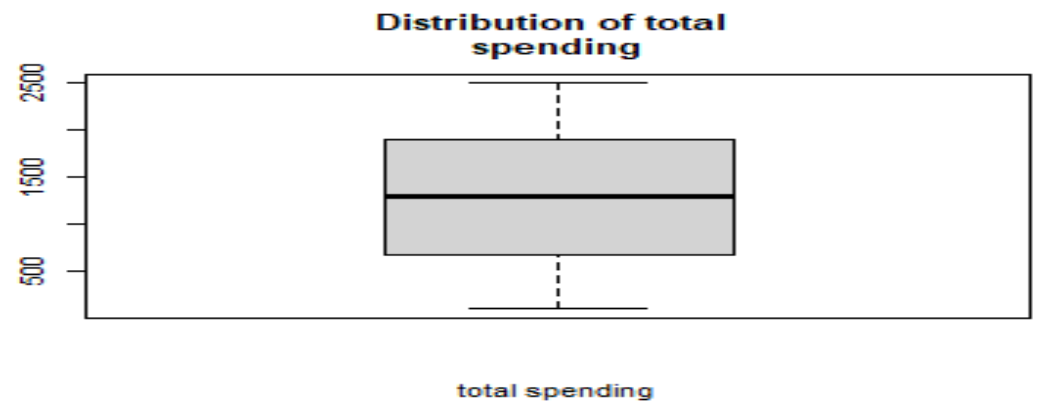

Compare cash and credit totals
totals
0
1000
3000
Cash
Credit
paymenttype


Compare cash and credit totals
Cash
Credit


totalspending
800000
1200000
1600000
30
40
50
60
age


Compare age and total spending
29
25
23
30
22
35
36
60
37
55
39
50


Compare age and total spending
Aswan
Cairo
Alexandria
Dakahlia
Sohag
Fayoum
Port Said
Gharbia
Giza
Hurghada


Distribution of total spending
500
1500
2500
total spending

D) KMEANS

We will use the built-in function:

```
groupedData<- group_by(grc,age)
groupedData<- summarise(groupedData,sum=sum(total))
numOfClusters <- (readline(prompt = "Enter the number of clusters from 2 to 4: "))
kmeans <- kmeans(groupedData,centers = numOfClusters)

if(numOfClusters >= 2 & numOfClusters <= 4){
print(kmeans)
}else
print("Wrong input")
customers<- grc$customer
ages<- grc$age
totals<- grc$total
clusterNumber<- kmeans$cluster

newtable<- data.frame(customers, ages, totals, clusterNumber)
colnames(newtable) <- c("customer","age","total","cluster number")
newtable
newtable<- data.frame(customers, ages, totals)
```

xdxdxd.R* | grc | Untitled1

Source on Save | Run | Source

```r
                                    #Splitting the customers according to the sum of total spending and their ages



groupedData<- group_by(grc,age)
groupedData<- summarise(groupedData,sum=sum(total))
numOfClusters <- (readline(prompt = "Enter the number of clusters from 2 to 4: "))
kmeans <- kmeans(groupedData,centers = numOfClusters)

if(numOfClusters >= 2 & numOfClusters <= 4){
  print(kmeans)
}else
 print("Wrong input")
customers<- grc$customer
ages<- grc$age
totals<- grc$total
clusterNumber<- kmeans$cluster

newtable<- data.frame(customers, ages, totals, clusterNumber)
colnames(newtable) <- c("customer","age","total","cluster number")
newtable
newtable<- data.frame(customers, ages, totals)

```

E) Apriori Algorithm

We will use the built-in function:

```
library(arules)
dataItems<-read.transactions("C:/Users/Enter Store/Downloads/grcDataset-master/grc.csv", sep=',')
inspect(dataItems)
minConfidence<- as.numeric(readline(" Enter confidence here from 0.001 to 1: "))
minSupport<- as.numeric(readline("Enter Support here from 0.001 to 1: "))
apriori_rules<- apriori(dataItems, parameter=list(support=minSupport, confidence=minConfidence,
minlen=2))
inspect(apriori_rules)
```

```
232
233
234
235                                          # Association rules alpriori algorithm
236
237 library(arules)
238 dataItems<-read.transactions("C:/Users/Enter Store/Downloads/grcDataset-master/grc.csv", sep=',')
239 inspect(dataItems)
240 minConfidence<- as.numeric(readline(" Enter confidence here from 0.001 to 1: "))
241 minSupport<- as.numeric(readline("Enter Support here from 0.001 to 1: "))
242 apriori_rules<- apriori(dataItems, parameter=list(support=minSupport, confidence=minConfidence, minlen=2))
243 inspect(apriori_rules)
244
245
246
247
```