Finolex Academy of Management & Technology, Ratnagiri

Department of MCA

Course:-MCAL13 Advanced Database Management System

Lab

Roll No: 40 **Name: Anish Ramakant Karlekar**

Practical – 06

Title: - Introduction to R Graphics and Data Importing

Aim: - To understand R Graphics and how to import data.

Lab Objectives: -

Students will understand following R programming concepts:

I. R Plotting

II. R Line

III. R – Scatterplots

IV. R – Pie Charts

V. Data Importing

Description: -

I. R Plotting

▷ **Plot**

- The plot() function is used to draw points (markers) in a diagram.
- The function takes parameters for specifying points in the diagram.
  - Parameter 1 specifies points on the x-axis. ■ Parameter 2 specifies points on the y-axis.
- At its simplest, you can use the plot() function to plot two numbers against each other:
- Example - Draw one point in the diagram, at position (1) and position (3):

```
plot(1, 3)
```

- To draw more points, use vectors:
- Example - Draw two points in the diagram, one at position (1, 3) and one in position (8,10):

```
plot(c(1, 8), c(3, 10))
```

▷ Multiple Points

○ You can plot as many points as you like, just make sure you have the same number of points in both axis:
○ Example

```
plot(c(1, 2, 3, 4, 5), c(3, 7, 8, 9, 12))
```

○ For better organization, when you have many values, it is better to use variables: ○ Example

```
x <- c(1, 2, 3, 4, 5)
y <- c(3, 7, 8, 9, 12) plot(x, y)
```

▷ Sequences of Points

○ If you want to draw dots in a sequence, on both the x-axis and the y-axis, use the : operator:
○ Example

```
plot(1:10)
```

II. R Line

▷ Line Graphs

○ A line chart is a graph that connects a series of points by drawing line segments between them.
○ These points are ordered in one of their coordinate (usually the x-coordinate) value.
○ Line charts are usually used in identifying the trends in data.
○ The plot() function in R is used to create the line graph. ○
Syntax - The basic syntax to create a line chart in R is −

```
plot(v,type,col,xlab,ylab)
```

○ Following is the description of the parameters used −

- v is a vector containing the numeric values.

Lab

- type takes the value "p" to draw only the points, "l" to draw only the lines and "o" to draw both points and lines.
- xlab is the label for x axis.
- ylab is the label for y axis.
- main is the Title of the chart.
- col is used to give colors to both the points and lines.

▷ **Line Graphs**

- ○ A line graph has a line that connects all the points in a diagram.
- ○ To create a line, use the plot() function and add the type parameter with a value of "l":
- ○ Example

```
plot(1:10, type="l")
```

▷ **Line Color**

- ○ The line color is black by default. To change the color, use the col parameter:
- ○ Example

```
plot(1:10, type="l", col="blue")
```

▷ **Line Width**

○ To change the width of the line, use the lwd parameter (1 is default, while 0.5 means 50% smaller, and 2 means 100% larger): ○ Example

```
plot(1:10, type="l", lwd=2)
```

▷ **Line Styles**

- ○ The line is solid by default. Use the lty parameter with a value from 0 to 6 to specify the line format.
- ○ For example, lty=3 will display a dotted line instead of a solid line:
- ○ Example

```
plot(1:10, type="l", lwd=5, lty=3)
```

- ○ Available parameter values for lty:

- 0 removes the line
- 1 displays a solid line
- 2 displays a dashed line
- 3 displays a dotted line
- 4 displays a "dot dashed" line
- 5 displays a "long dashed" line
- 6 displays a "two dashed" line

⮚ **Multiple Lines**

- More than one line can be drawn on the same chart by using the lines()function.
- After the first line is plotted, the lines() function can use an additional vector as input to draw the second line in the chart,
- To display more than one line in a graph, use the plot() function together with the lines() function:
- Example

```
line1 <- c(1,2,3,4,5,10) line2 <-
c(2,5,7,8,9,10) plot(line1, type =
"l", col = "blue") lines(line2,
type="l", col = "red")
```

III. R - Scatterplots

▷ **R - Scatterplots**

- Scatterplots show many points plotted in the Cartesian plane.
- Each point represents the values of two variables.
- One variable is chosen in the horizontal axis and another in the vertical axis.

○ The simple scatterplot is created using the plot() function. ○ Syntax - The basic syntax for creating scatterplot in R is − plot(x, y, main, xlab, ylab, xlim, ylim, axes)

▷ Example

```
x <- c(5,7,8,7,2,2,9,4,11,12,9,6)
y <- c(99,86,87,88,111,103,87,94,78,77,85,86)
  plot(x, y, main="Observation of Cars", xlab="Car age", ylab="Car speed")
```

- The observation in the example above should show the result of 12 cars passing by.

Lab

- The x-axis shows how old the car is.
- The y-axis shows the speed of the car when it passes.

▷ **Compare Plots**

- To compare the plot with another plot, use the points() function:
- Example - Draw two plots on the same figure:

```
# day one, the age and speed of 12 cars:
x1 <- c(5,7,8,7,2,2,9,4,11,12,9,6) y1 <-
c(99,86,87,88,111,103,87,94,78,77,85,86) #
day two, the age and speed of 15 cars:
x2 <- c(2,2,8,1,15,8,12,9,7,3,11,4,7,14,12)
y2 <- c(100,105,84,105,90,99,90,95,94,100,79,112,91,80,85)
plot(x1, y1, main="Observation of Cars", xlab="Car age", ylab="Car speed", col="red",
cex=2) points(x2, y2, col="blue", cex=2)
```

IV. R - Pie Charts

▷ **Pie Charts**

- A pie chart is a circular graphical view of data. In R the pie chart is created using the pie() function which takes positive numbers as a vector input. The additional parameters are used to control labels, color, title etc.
- Syntax - The basic syntax for creating a pie-chart using the R is − pie(x, labels, radius, main, col, clockwise)

- Following is the description of the parameters used −
  - x is a vector containing the numeric values used in the pie chart.
  - labels is used to give description to the slices.
  - radius indicates the radius of the circle of the pie chart.(value between −1 and +1).
  - main indicates the title of the chart.
  - col indicates the color palette.
  - clockwise is a logical value indicating if the slices are drawn clockwise or anti clockwise.
- Example

```
# Create a vector of
pies x <- c(10,20,30,40)
# Display the pie chart
pie(x)
```

Lab

- As you can see the pie chart draws one pie for each value in the vector (in this case 10, 20, 30, 40).
- By default, the plotting of the first pie starts from the x-axis and move counterclockwise.
- Note: The size of each pie is determined by comparing the value with all the other values, by using this formula:
- The value divided by the sum of all values: x/sum(x)

▷ **Labels and Header**

- Use the label parameter to add a label to the pie chart, and use the main parameter to add a header: ○ Example

```
# Create a vector of pies x
<- c(10,20,30,40) #
Create a vector of labels
mylabel <- c("Apples", "Bananas", "Cherries", "Dates")
# Display the pie chart with labels
pie(x, label = mylabel, main = "Fruits")
```

▻ **Colors**

- You can add a color to each pie with the col parameter:
- Example

```
# Create a vector of colors
colors <- c("blue", "yellow", "green", "black")
# Display the pie chart with colors
pie(x, label = mylabel, main = "Fruits", col = colors)
```

▻ **Legend**

- To add a list of explanation for each pie, use the legend() function:
- The legend can be positioned as either: bottomright, bottom, bottomleft, left, topleft, top, topright, right, center
- Example

```
# Create a vector of pies x
<- c(10,20,30,40) #
Create a vector of labels
mylabel <- c("Apples", "Bananas", "Cherries", "Dates")
# Create a vector of colors
colors <- c("blue", "yellow", "green", "black")
# Display the pie chart with colors
pie(x, label = mylabel, main = "Pie Chart", col = colors)
```

```
# Display the explanation box
legend("bottomright", mylabel, fill = colors)
```

V. Importing Data

▷ In R, we can read data from files stored outside the R environment.
▷ We can also write data into files which will be stored and accessed by the operating system.
▷ R can read and write into various file formats like csv, excel, xml etc.
▷ The file should be present in current working directory so that R can read it.
▷ Of course we can also set our own directory and read files from there.

Importing data from CSV files

▷ The csv file is a text file in which the values in the columns are separated by a comma.

▷ **Reading a CSV File**

- Following is a simple example of read.csv() function to read a CSV file available in your current working directory −

```
data <- read.csv("input.csv")
print(data)
```

▻ **Get Information**

- Use the dim() function to find the dimensions of the data set, and the names() function to view the names of the variables:

```
dim(data)
names(data)
rownames(data)
```

- By default, the read.csv() function gives the output as a data frame.

▻ **Access the data frame by using the $ sign, and the name of the variable.**

```
data$name
```

Lab

- Once we read data in a data frame, we can apply all the functions applicable to data frames
- Example - Get the maximum salary

```
# Get the max salary from data frame.
sal <- max(data$salary)
print(sal)
```

Sort data

- To sort the values, use the sort() function:

```
sort(data$salary)
```

Analyzing the Data

- Now that we have some information about the data set, we can start to analyze it with some statistical numbers.
- For example, we can use the summary() function to get a statistical summary of the data:
- Example

```
Data_Cars <- mtcars
summary(Data_Cars)
```

- The summary() function returns six statistical numbers for each variable:
  - Min
  - First quantile (percentile)
  - Median
  - Mean
  - Third quantile (percentile)
  - Max

Lab

⮚ **Some Examples of accessing data**

- ○ Get the details of the person with max salary
- ○ We can fetch rows meeting specific filter criteria similar to a SQL where clause.

```
# Create a data frame.
data <- read.csv("input.csv")
# Get the max salary from data frame.
sal <- max(data$salary)
# Get the person detail having max salary.
retval <- subset(data, salary == max(salary))
print(retval)
```

▷ Get all the people working in IT department

```
# Create a data frame.
data <- read.csv("input.csv")
details <- subset( data, dept == "IT")
print(details)
```

▷ Get the persons in IT department whose salary is greater than 600

```
# Create a data frame.
data <- read.csv("input.csv")
info <- subset(data, salary > 600 & dept == "IT")
print(info)
```

▷ Get the people who joined on or after 2014

```
# Create a data frame.
data <- read.csv("input.csv")
info <- subset(data, as.Date(start_date) > as.Date("2014-01-01"))
print(info)
```

▷ **Writing into a CSV File**

- ○ R can create csv file form existing data frame.
- ○ The write.csv() function is used to create the csv file. This file gets created in the working directory.

```
# Create a data frame.
data <- read.csv("input.csv") retval <- subset(data,
as.Date(start_date) > as.Date("2014-01-01")) # Write filtered data
```

into a new file. write.csv(retval,"output.csv") newdata <-
read.csv("output.csv") print(newdata)

Importing Data - Excel File

▷ Microsoft Excel is the most widely used spreadsheet program which stores data in the .xls or .xlsx format.
▷ R can read directly from these files using some excel specific packages.
▷ Few such packages are - XLConnect, xlsx, gdata etc.
▷ We will be using xlsx package. R can also write into excel file using this package.
▷ Install xlsx Package
- ○ You can use the following command in the R console to install the "xlsx" package.
- ○ It may ask to install some additional packages on which this package is dependent.
- ○ Follow the same command with required package name to install the additional packages.

```
install.packages("xlsx")
```

▷ Verify and Load the "xlsx" Package
- ○ Use the following command to verify and load the "xlsx" package.
  - ■ # Verify the package is installed.
  - ■ any(grepl("xlsx",installed.packages())) ○

Load the library into R workspace.
- ■ library("xlsx")
- ■ print(library("xlsx"))

▷ **Input as xlsx File**
- ○ You should save it in the current working directory of the R workspace.

▷ **Reading the Excel File**
- ○ The input.xlsx is read by using the read.xlsx() function as shown below. The result is stored as a data frame in the R environment.

```
# Read the first worksheet in the file input.xlsx.
data <- read.xlsx("input.xlsx", sheetIndex = 1)
print(data)
```

▷ **Writing data into Excel File**
- ○ In R, we can also write the data into our .xlsx file.
- ○ R provides a write.xlsx() function to write data into the excel file. ○

There is the following syntax of write.xlsx() function:

```
write.xlsx(data_frame,file_name,col.names,row.names,sheetnames,append)
```

Lab

- The data_frame is our data, which we want to insert into our excel file.
- The file_names is the name of that file in which we want to insert our data.
- The col.names and row.names are the logical values that are specifying whether the column names/row names of the data frame are to be written to the file.
- The append is a logical value, which indicates our data should be appended or not into an existing file.

- Example

```
# Create a data frame.
data <- read.xlsx("input.xlsx", sheetIndex = 1) empdata <-
subset(data, as.Date(start_date) > as.Date("2014-01-01")) # Write
filtered data into a new file.
write.xlsx(empdata,"emp.xlsx",col.names=TRUE,
row.names=TRUE,sheetName="Sheet2",append = TRUE) newdata <-
read.xlsx("emp.xlsx",sheetIndex = 1) print(newdata)
```

Lab

Exercises

1. **Import employee.csv file and perform following -**

Display the content.

```
R Console
> setwd("D:/MCA/ADBMS/prac_06")
> getwd()
[1] "D:/MCA/ADBMS/prac_06"
> data <- read.csv("employee.csv")
> print(data);
   id     Name Age     Designation Salary isLocal
1   1 Michelle  44         Manager  72000      NA
2   2     Ryan  27           Clerk  48000      NA
3   3     Gary  30           Clerk  54000      NA
4   4     Guru  38        Engineer  61000      NA
5   5    Harsh  40           Clerk     NA      NA
6   6     Brad  35        Engineer  58000      NA
7   7    James  NA           Clerk  52000      NA
8   8     Tina  48 Senior_manager  79000      NA
9   9     Mina  50             CEO  83000      NA
10 10     Tara  37        Engineer  67000      NA
>
>
```

2. Find the dimensions of the data in the above imported dataset.

```
>
> dim(data)
[1] 10  6
>
>
```

3. Get all the people with designation “clerk”.

```
>
> Clerk <- subset( data, Designation == "Clerk")
> print(Clerk)
  id  Name Age Designation Salary isLocal
2  2  Ryan  27       Clerk  48000      NA
3  3  Gary  30       Clerk  54000      NA
5  5 Harsh  40       Clerk     NA      NA
7  7 James  NA       Clerk  52000      NA
>
>
```

4. Get the people whose salary is greater than 55,000 and write the output in new excel file.

Lab

```
>
> amount <- subset(data, Salary >= 55000)
> print(amount)
   id     Name Age    Designation Salary isLocal
1   1 Michelle  44        Manager  72000      NA
4   4      Guru  38       Engineer  61000      NA
6   6      Brad  35       Engineer  58000      NA
8   8      Tina  48 Senior_manager  79000      NA
9   9      Mina  50            CEO  83000      NA
10 10      Tara  37       Engineer  67000      NA
> write.csv(amount,"output.csv")
> newdata <- read.csv("output.csv")
> print(newdata)
   X id     Name Age    Designation Salary isLocal
1  1  1 Michelle  44        Manager  72000      NA
2  4  4      Guru  38       Engineer  61000      NA
3  6  6      Brad  35       Engineer  58000      NA
4  8  8      Tina  48 Senior_manager  79000      NA
5  9  9      Mina  50            CEO  83000      NA
6 10 10      Tara  37       Engineer  67000      NA
>
>
```

5. Summarize the above dataset

```
>
> summary(newdata)
       X                id              Name                Age
 Min.   : 1.000   Min.   : 1.000   Length:6           Min.   :35.00
 1st Qu.: 4.500   1st Qu.: 4.500   Class :character   1st Qu.:37.25
 Median : 7.000   Median : 7.000   Mode  :character   Median :41.00
 Mean   : 6.333   Mean   : 6.333                      Mean   :42.00
 3rd Qu.: 8.750   3rd Qu.: 8.750                      3rd Qu.:47.00
 Max.   :10.000   Max.   :10.000                      Max.   :50.00
 Designation            Salary      isLocal
 Length:6           Min.   :58000   Mode:logical
 Class :character   1st Qu.:62500   NA's:6
 Mode  :character   Median :69500
                    Mean   :70000
                    3rd Qu.:77250
                    Max.   :83000
>
>
```

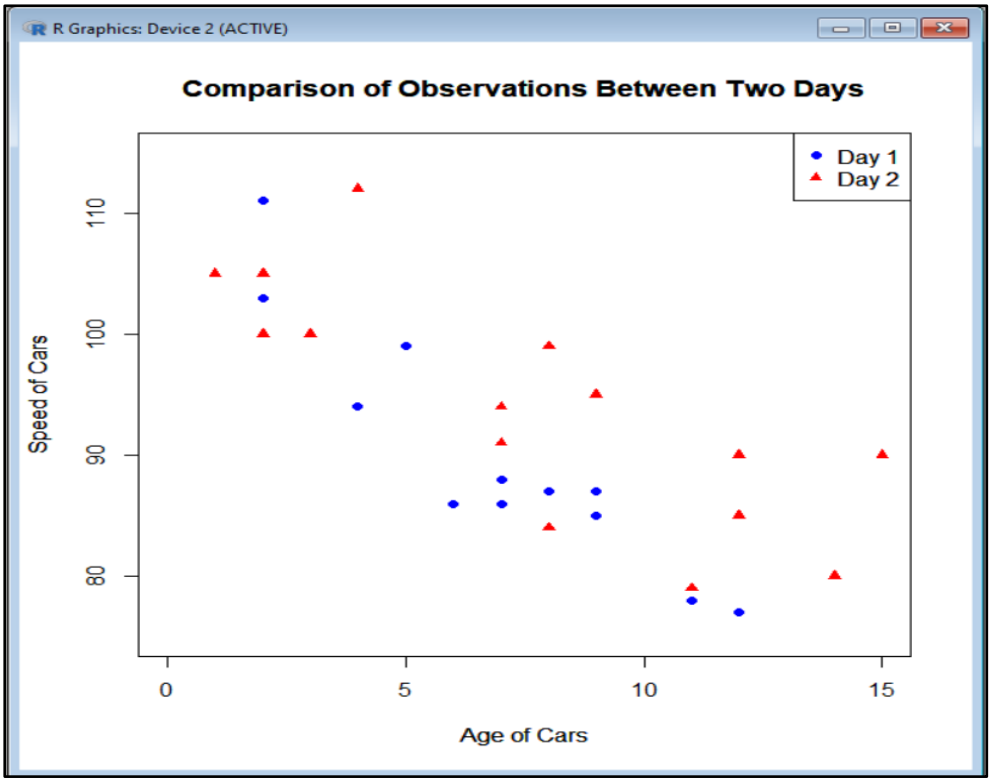
Lab

2. **The age and speed of 12 cars observed on day 1 are age1 (5,7,8,7,2,2,9,4,11,12,9,6), speed1 (99,86,87,88,111,103,87,94,78,77,85,86) and on day 2 following values are observed age2(2,2,8,1,15,8,12,9,7,3,11,4,7,14,12), speed2(100,105,84,105,90,99,90,95,94,100,79,112,91,80,85). Write a R program to draw a scatterplot that compares observations of the two days.**

CODE :

```
age1 <- c(5, 7, 8, 7, 2, 2, 9, 4, 11, 12, 9, 6)

speed1 <- c(99, 86, 87, 88, 111, 103, 87, 94, 78, 77, 85, 86)

age2 <- c(2, 2, 8, 1, 15, 8, 12, 9, 7, 3, 11, 4, 7, 14, 12)

speed2 <- c(100, 105, 84, 105, 90, 99, 90, 95, 94, 100, 79, 112, 91, 80, 85)

plot(age1, speed1, col = "blue", pch = 16, xlab = "Age of Cars", ylab = "Speed of Cars",

    main = "Comparison of Observations Between Two Days", xlim = c(0, 15), ylim = c(75, 115))points(age2, speed2, col = "red", pch = 17)

legend("topright", legend = c("Day 1", "Day 2"), col = c("blue", "red"), pch = c(16, 17))
```

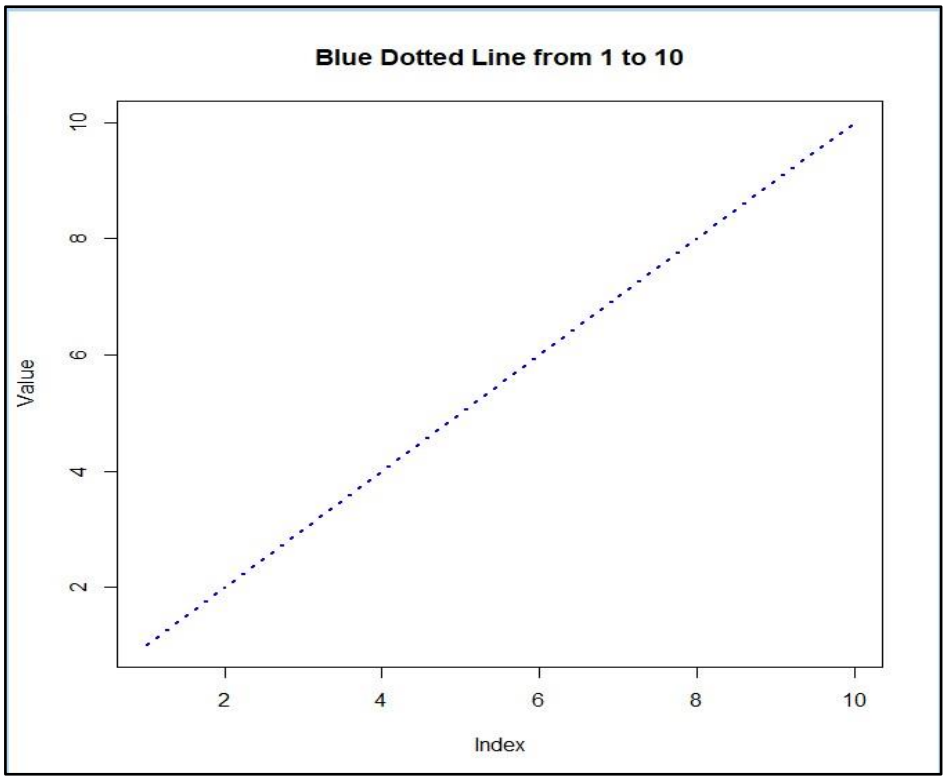
OUTPUT :



Lab

3. **Write a R program to create a vector with numerical values in a sequence from 1 to 10 and draw a blue colored dotted line of width 2 for the above vector.**

```
> v <- 1:10
> plot(v, type="l", col="blue", lty=3, lwd=2, xlab="Index", ylab="Value",
+   main="Blue Dotted Line from 1 to 10")
```



4. **Write a R program to read the excel file "input.xlsx" and perform following**

1. Display the content.

```
> data <- read.xlsx("input.xlsx", sheetIndex=1)
> print(data)
  id      name salary start_date       dept
1  1      Rick 623.30 2012-01-01         IT
2  2       Dan 515.20 2013-09-23 Operations
3  3 Michelle 611.00 2014-11-15         IT
4  4      Ryan 729.00 2014-05-11         HR
5  5      Gary 843.25 2015-03-27    Finance
6  6      Nina 578.00 2013-05-21         IT
7  7     Simon 632.80 2013-07-30 Operations
8  8      Guru 722.50 2014-06-17    Finance
```

2. Find the dimensions of the data in the above imported dataset.

```
> dim(data)
[1] 8 5
```

Lab

3. Get all the people working in IT department

```
> it_data <- subset(data, dept == "IT")
> print(it_data)
  id      name salary start_date dept
1  1      Rick  623.3 2012-01-01   IT
3  3 Michelle  611.0 2014-11-15   IT
6  6      Nina  578.0 2013-05-21   IT
```

4. Get the people who joined on or after 2014 and write the output in new excel file.

```
> joined_after_2014 <- subset(data, as.Date(start_date) >= as.Date("2014-01-01"))
> write.xlsx(joined_after_2014, "joined_after_2014.xlsx")
```

Joined_after_2014.xlsx

	A	B	C	D	E	F
1		id	name	salary	start_date	dept
2	3	3	Michelle	611	11/15/2014	IT
3	4	4	Ryan	729	5/11/2014	HR
4	5	5	Gary	843.25	3/27/2015	Finance
5	8	8	Guru	722.5	6/17/2014	Finance

5. Summarize the above dataset

```
> summary(data)
       id           name               salary       start_date            dept
 Min.   :1.00   Length:8           Min.   :515.2   Min.   :2012-01-01   Length:8
 1st Qu.:2.75   Class :character   1st Qu.:602.8   1st Qu.:2013-07-12   Class :character
 Median :4.50   Mode  :character   Median :628.0   Median :2014-01-16   Mode  :character
 Mean   :4.50                      Mean   :656.9   Mean   :2013-12-13
 3rd Qu.:6.25                      3rd Qu.:724.1   3rd Qu.:2014-07-24
 Max.   :8.00                      Max.   :843.2   Max.   :2015-03-27
```

Lab

5.Create a pie chart for favourite movie categories (comedy,action,drama,romance,sci-fi). Consider appropriate percentages for creating pies. Add a list of explanation for each pie

```
> categories <- c("Comedy", "Action", "Drama", "Romance", "Sci-Fi")
> percentages <- c(20, 30, 25, 15, 10)  # Appropriate percentages
>
>
> pie(percentages, labels=categories, main="Favorite Movie Categories",
+ col=c("blue", "red", "green", "yellow", "purple"))
> legend("topright", categories, fill=c("blue", "red", "green", "yellow", "purple"))
>
```