Program No.: 1

Aim: Introduction to R in Data Science.

Theory:

R is a powerful and versatile open-source programming language primarily designed for statistical computing, data analysis, and visualization. It was created in the mid-1990s by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand. Since then, R has evolved into one of the most widely used tools for researchers, data analysts, and statisticians worldwide.

Key Characteristics of R:

1. **Statistical Computing:**
   R provides a rich environment for statistical techniques such as hypothesis testing, regression, clustering, classification, and time-series analysis.
2. **Data Manipulation:**
   With built-in data structures like vectors, matrices, data frames, and lists, R allows efficient storage, transformation, and analysis of data. Packages like dplyr and tidyr further enhance these capabilities.
3. **Data Visualization:**
   R has powerful tools for data visualization. Packages such as ggplot2 and lattice can generate simple charts to complex, multi-layered graphics that aid in understanding trends and patterns in data.
4. **Extensibility:**
   R is open-source and supports thousands of user-contributed packages, which extend its functionality for specialized tasks like machine learning (caret), deep learning (keras), geospatial analysis (sf), and bioinformatics (Bioconductor).

5. **Cross-Platform & Integration:**
   R can run on Windows, macOS, and Linux. It integrates with databases (SQL, MongoDB), other languages (Python, Java, C++), and big data tools (Hadoop, Spark).
6. **Community & Support:**
   R has a large global community of developers and researchers. The CRAN (Comprehensive R Archive Network) hosts over 20,000 packages, ensuring support for a wide range of analytical tasks.

Applications of R in Data Science:

- **Data Cleaning & Preprocessing:** Handling missing values, filtering, and transforming data.
- **Statistical Analysis:** Performing tests, building models, and making predictions.
- **Visualization & Reporting:** Creating interactive dashboards and reports.
- **Machine Learning:** Implementing classification, regression, clustering, and neural networks.
- **Research & Academia:** R is widely used in scientific research for reproducible results.

In summary, R is not just a programming language but a complete environment for data-driven decision-making. Its combination of statistical power, visualization capabilities, and package ecosystem makes it a preferred choice in Data Science.

Program No.: 2

Aim: To study the installation of R and explore its basic syntax.

Theory:

R can be installed from CRAN (Comprehensive R Archive Network). After installation, RStudio (an IDE for R) can be used for writing and running R programs.

Some **basic syntax elements of R** are:

1. **Variables and Assignment:** Values can be assigned using <- or =.
2. **Data Types:** R supports numeric, character, logical, complex, and raw data types.
3. **Vectors:** A one-dimensional collection of data elements, created using c().
4. **Operators:** Arithmetic (+, -, *, /), relational (<, >, ==), and logical (&, |).
5. **Functions:** Built-in and user-defined functions are used to perform operations.

Source Code:

```
# Assigning values

x <- 10

y = 20

print(x)

print(y)

# Creating a vector

vec <- c(1, 2, 3, 4, 5)

print(vec)

# Arithmetic operations

sum <- x + y
```

product <- x * y

print(sum)

print(product)

Logical comparison

result <- x < y

print(result)

Built-in function

sqrt_val <- sqrt(25)

print(sqrt_val)

Output:

```
[1] 10
[1] 20
[1] 1 2 3 4 5
[1] 30
[1] 200
[1] TRUE
[1] 5
```

Program No.: 3

Aim: To describe data, and perform viewing and manipulation operations using R.

Theory:

R provides powerful tools for data description and manipulation. Data can be imported, explored, and transformed efficiently.

- **Describing Data:** Functions like summary(), str(), and dim() provide insights into datasets.
- **Viewing Data:** The head(), tail(), and View() functions allow quick inspection.
- **Manipulating Data:** Using indexing, filtering, and packages like dplyr for operations such as select(), filter(), mutate(), arrange().

Source Code:

```
# Creating a sample data frame
data <- data.frame(
  ID = 1:5,
  Name = c("A", "B", "C", "D", "E"),
  Age = c(21, 25, 19, 23, 22),
  Score = c(85, 90, 78, 88, 95)
)
# Describing the data
print("Summary of Data:")
print(summary(data))
print("Structure of Data:")
```

print(str(data))

Viewing data

print("First few rows:")

print(head(data, 3))

print("Last few rows:")

print(tail(data, 2))

Manipulating data

print("Selecting Age column:")

print(data$Age)

print("Filtering rows where Score > 85:")

print(subset(data, Score > 85))

Output:

```
[1] "Summary of Data:"
       ID         Name                 Age          Score     
 Min.   :1   Length:5           Min.   :19   Min.   :78.0  
 1st Qu.:2   Class :character   1st Qu.:21   1st Qu.:85.0  
 Median :3   Mode  :character   Median :22   Median :88.0  
 Mean   :3                      Mean   :22   Mean   :87.2  
 3rd Qu.:4                      3rd Qu.:23   3rd Qu.:90.0  
 Max.   :5                      Max.   :25   Max.   :95.0  
[1] "Structure of Data:"
'data.frame':   5 obs. of  4 variables:
 $ ID   : int  1 2 3 4 5
 $ Name : chr  "A" "B" "C" "D" ...
```

```
 $ Age  : num  21 25 19 23 22
 $ Score: num  85 90 78 88 95
NULL
[1] "First few rows:"
  ID Name Age Score
1  1    A  21    85
2  2    B  25    90
3  3    C  19    78
[1] "Last few rows:"
  ID Name Age Score
4  4    D  23    88
5  5    E  22    95
```

```
[1] "Selecting Age column:"
[1] 21 25 19 23 22
[1] "Filtering rows where Score > 85:"
  ID Name Age Score
2  2    B  25    90
4  4    D  23    88
5  5    E  22    95
```

Program No.: 4

Aim: To visualize data using tables, charts, and plots in R.

Theory:

R is widely known for its strong visualization capabilities. Some methods include:

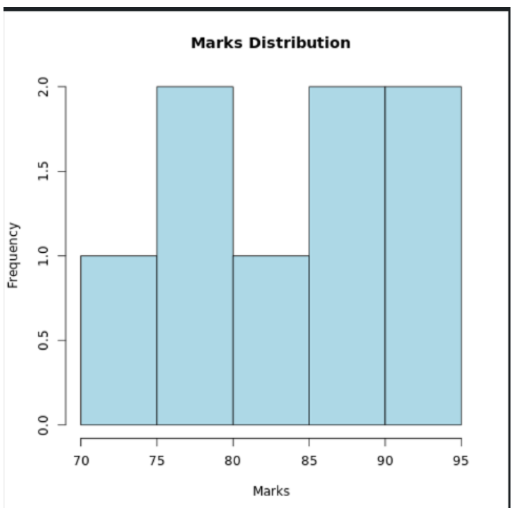
- **Tables:** Using table() for frequency distributions.
- **Charts and Plots:** Functions like barplot(), hist(), pie(), and plot().
- **Advanced Visualizations:** Using ggplot2 for layered and customizable graphics.

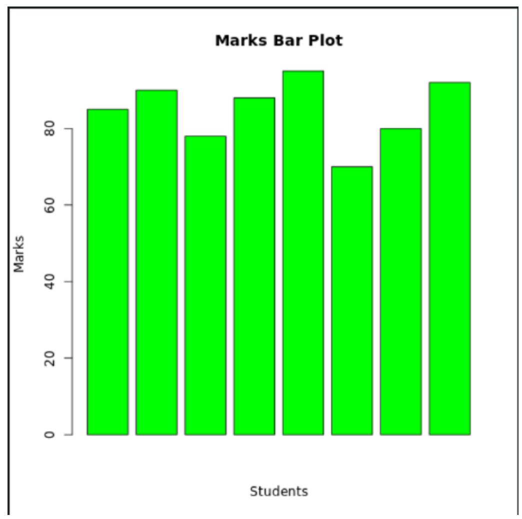
Source Code:

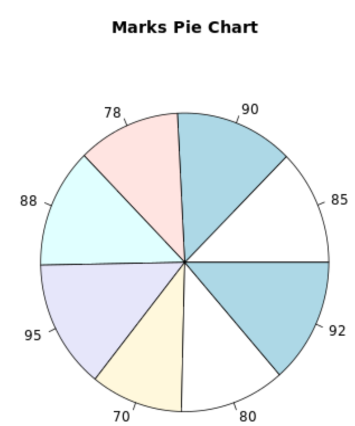
```
marks <- c(85, 90, 78, 88, 95, 70, 80, 92)

# Frequency table

print("Frequency Table:")

print(table(marks))

# Histogram

hist(marks, col="lightblue", main="Marks Distribution", xlab="Marks")

# Barplot

barplot(marks, col="green", main="Marks Bar Plot", xlab="Students", ylab="Marks")

# Pie chart

pie(marks, labels = marks, main="Marks Pie Chart")

# Scatter plot

plot(marks, type="o", col="blue", main="Scatter Plot of Marks", xlab="Index", ylab="Marks")
```

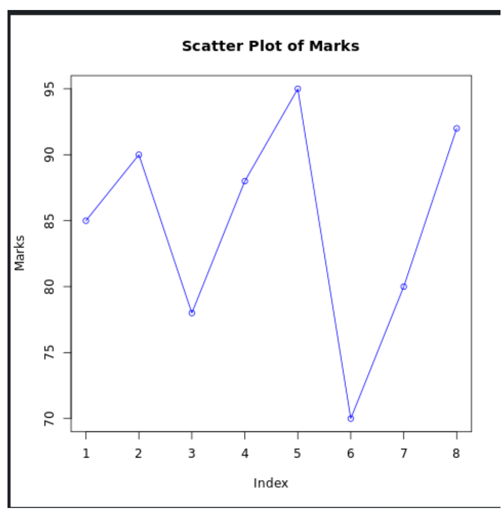
Output:

```
[1] "Frequency Table:"
marks
70 78 80 85 88 90 92 95
 1  1  1  1  1  1  1  1
```









Program No.: 5

Aim: To implement binomial distribution and plot its density and distribution functions in R.

Theory:

The **Binomial distribution** is a discrete probability distribution representing the number of successes in a fixed number of independent trials, each with the same probability of success.

- Probability Mass Function: $P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$
- In R:
  - dbinom(k, n, p) → Probability at k.
  - pbinom(k, n, p) → Cumulative probability up to k.
  - rbinom(n, size, prob) → Generates random binomial values.

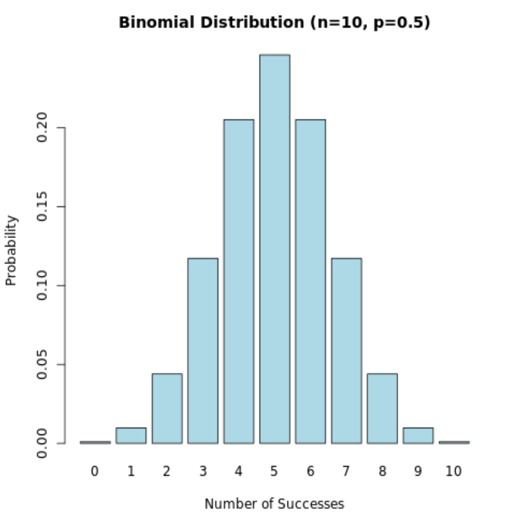
Source Code:

```
# Parameters

n <- 10   # number of trials

p <- 0.5  # probability of success

# Probability distribution

x <- 0:n

prob <- dbinom(x, n, p)

# Print probabilities

print("Binomial Probabilities:")

print(prob)

# Plotting probability mass function
```

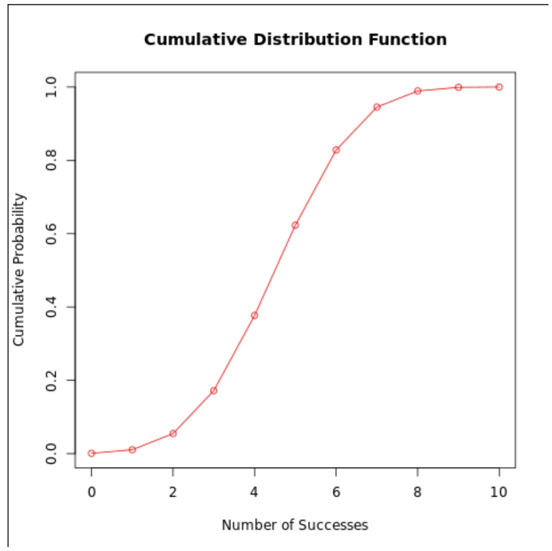
```r
barplot(prob, names.arg=x, col="lightblue",
        main="Binomial Distribution (n=10, p=0.5)",
        xlab="Number of Successes", ylab="Probability")
# Cumulative distribution
cum_prob <- pbinom(x, n, p)
print("Cumulative Probabilities:")
print(cum_prob)
# Plot cumulative distribution
plot(x, cum_prob, type="o", col="red",
     main="Cumulative Distribution Function",
     xlab="Number of Successes", ylab="Cumulative Probability")
```

Output:

```
[1] "Binomial Probabilities:"
 [1] 0.0009765625 0.0097656250 0.0439453125 0.1171875000 0.2050781250
 [6] 0.2460937500 0.2050781250 0.1171875000 0.0439453125 0.0097656250
[11] 0.0009765625
[1] "Cumulative Probabilities:"
 [1] 0.0009765625 0.0107421875 0.0546875000 0.1718750000 0.3769531250
 [6] 0.6230468750 0.8281250000 0.9453125000 0.9892578125 0.9990234375
[11] 1.0000000000
```



Binomial Distribution (n=10, p=0.5)
Probability
0.00
0.05
0.10
0.15
0.20
0
1
2
3
4
5
6
7
8
9
10
Number of Successes



Cumulative Distribution Function
Cumulative Probability
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
Number of Successes