Date – 13/01/2025

Practical No : 1

AIM – Installation of R and R Studio

Installing R, RStudio and various R packages

This handout describes the installation of R, RStudio and various R packages.

I) Install R

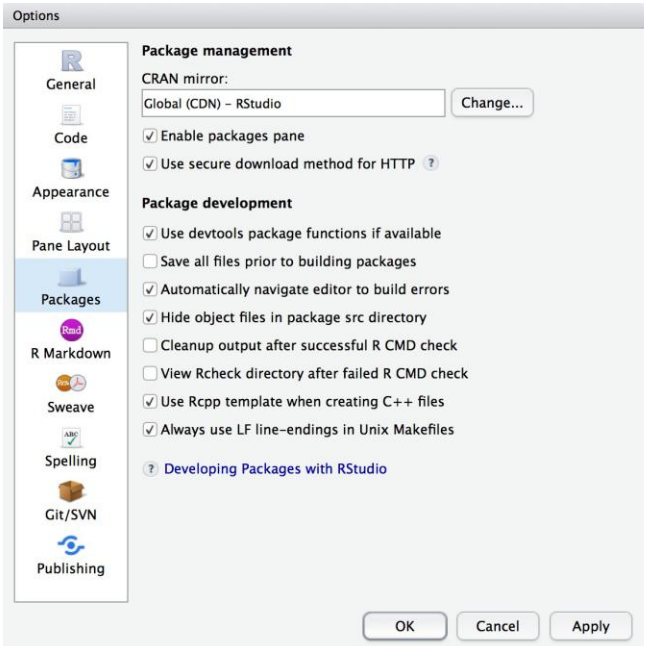
1. Go to the site of the Comprehensive R Archive Network: https://cran.r-project.org
2. Click the Download button for your system.
3. Select the installation file for your system. Windows users should select the 'base' distribution.
4. Run the installation file.

II) Install RStudio (graphical user interface for R)

1. Go to the RStudio Download page: https://posit.co/products/
2. Click the Download RStudio Desktop button.
3. Select the installation file for your system.
4. Run the installation file.

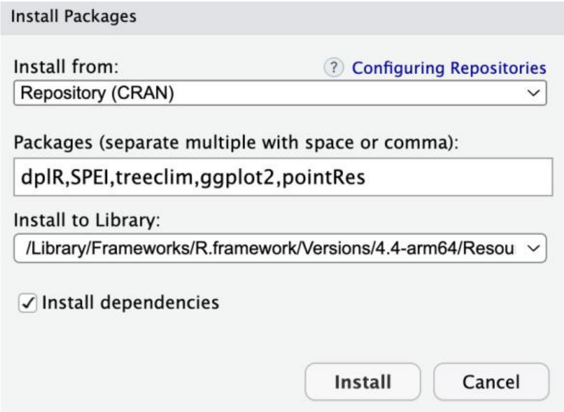
III) Check whether a CRAN mirror is selected

1. Open RStudio.
2. For Windows: Click Tools > Global Options > Packages.
3. Check whether a mirror is selected under 'CRAN mirror'. If not, specify any mirror via the Change button.



IV) Install the packages dplR, SPEI, treeclim, ggplot2, pointRes

1. Go to the Packages tab in the bottom right panel of RStudio.
2. Click the Install packages button.
3. Select install from 'Repository (CRAN)', type the package names separated by a comma, and tick 'install dependencies'.
4. Click the Install button.



Date – 20/01/2025

Practical No : 2

AIM – To study about different types of operators in R

1) Arithmetic operations

```
var.add = var.1 + var.2
var.sub = var.1 - var.2
var.mul = var.1 * var.2
var.div = var.1 / var.2
var.mod = var.1 %% var.2
var.quot = var.1 %/% var.2
cat("Addition of two vectors:", var.add, "\n")
cat("Subtraction of two vectors:", var.sub, "\n")
cat("Multiplication of two vectors:", var.mul, "\n")
cat("Division of two vectors:", var.div, "\n")
cat("Modulus of two vectors:", var.mod, "\n")
cat("Quotient of two vectors:", var.quot, "\n")
```

```
Addition of two vectors: 4 6 8 10
Subtraction of two vectors: -4 -4 -4 -4
Multiplication of two vectors: 0 5 12 21
Division of two vectors: 0 0.2 0.3333333 0.4285714
Modulus of two vectors: 0 1 2 3
Quotient of two vectors: 0 0 0 0
```

2) Relational operations

```
var.great = var.1 > var.2
var.less = var.1 < var.2
var.equal = var.1 == var.2
var.greatequal= var.1 >= var.2
var.lessequal = var.1 <= var.2
cat("Greater than comparison of two vectors:", var.great, "\n")
cat("Less than comparison of two vectors:", var.less, "\n")
cat("Equal to comparison of two vectors:", var.equal, "\n")
cat("Greater than or equal to comparison of two vectors:", var.greatequal, "\n")
```

```
Greater than comparison of two vectors: FALSE FALSE FALSE FALSE
Less than comparison of two vectors: TRUE TRUE TRUE TRUE
Equal to comparison of two vectors: FALSE FALSE FALSE FALSE
Greater than or equal to comparison of two vectors: FALSE FALSE FALSE FALSE
Less than or equal to comparison of two vectors: TRUE TRUE TRUE TRUE
```

```
cat("Less than or equal to
comparison of two vectors:",
var.lessequal, "\n")
```

3) Logical operations

```
var.and = var.1 & var.2
var.or = var.1 | var.2
var.not = !var.1
var.aand = var.1[1] &&
var.2[1]
cat("AND operation of two
vectors:", var.and, "\n")
cat("OR operation of two
vectors:", var.or, "\n")
cat("NOT operation of two
vectors:", var.not, "\n")
cat("AND operation of two
vectors:", var.aand, "\n")
```

```
AND operation of two vectors: FALSE TRUE TRUE TRUE
OR operation of two vectors: TRUE TRUE TRUE TRUE
NOT operation of two vectors: TRUE FALSE FALSE FALSE
AND operation of two vectors: FALSE
```

4) Assignment operations

```
var.1 = var.2
cat("Assignment of two
vectors:", var.1, "\n")
```

```
Assignment of two vectors: 4 5 6 7
```

5) Miscellaneous operations

```
var.seq = seq(1, 10, by = 2)
var.rep = rep(1, 10)
var.rev = rev(var.1)
var.sort = sort(var.1)
cat("Sequence of numbers:",
var.seq, "\n")
cat("Repeated numbers:",
var.rep, "\n")
cat("Reversed numbers:",
var.rev, "\n")
cat("Sorted numbers:",
var.sort, "\n")
```

```
Sequence of numbers: 1 3 5 7 9
Repeated numbers: 1 1 1 1 1 1 1 1 1 1
Reversed numbers: 3 2 1 0
Sorted numbers: 0 1 2 3
```

6) Vector

```
v= c(1,2,3,4,5)
cat("Vector:", v, "\n")
cat("Vector:", seq(5,9, by =
0.4), "\n")
v1 = 6.6:12.6
cat("Vector:", v1, "\n")
v=c("Monday", "Tuesday",
"Wednesday", "Thursday",
"Friday", "Saturday",
"Sunday")
cat("Day:", v[1], "\n")
b=v[c(2,3,6)]
cat("Days:", b, "\n")
v=c(0,1,3,5,10,-1,-2,-3)
cat("Vector:", v, "\n")
cat("Sorted Vector:", sort(v),
"\n")
```

```
Vector: 1 2 3 4 5
Vector: 5 5.4 5.8 6.2 6.6 7 7.4 7.8 8.2 8.6 9
Vector: 6.6 7.6 8.6 9.6 10.6 11.6 12.6
Day: Monday
Days: Tuesday Wednesday Saturday
Vector: 0 1 3 5 10 -1 -2 -3
Sorted Vector: -3 -2 -1 0 1 3 5 10
```

Date – 27/01/2025

Practical No : 3

AIM – To study about matrices, CSV, XLSX in R

1) Create a matrix in which elements are arranged sequentially by row and col

mrow=matrix(c(3:14), nrow=3, ncol=4, byrow=TRUE)
mcol=matrix(c(3:14), nrow=3, ncol=4, byrow=FALSE)
print(mrow)
print(mcol)

```
     [,1] [,2] [,3] [,4]
[1,]    3    4    5    6
[2,]    7    8    9   10
[3,]   11   12   13   14
     [,1] [,2] [,3] [,4]
[1,]    3    6    9   12
[2,]    4    7   10   13
[3,]    5    8   11   14
```

2) Define the row and column names of the matrix

rownames(mrow)=c("row1", "row2", "row3")
colnames(mrow)=c("col1", "col2", "col3", "col4")
print(mrow)

```
     col1 col2 col3 col4
row1    3    4    5    6
row2    7    8    9   10
row3   11   12   13   14
```

3) Create and Read CSV file in R

id,name,salary,start_date,department
1,Alice,50000,2020-03-15,HR
2,Bob,60000,2019-07-20,Finance
3,Charlie,75000,2021-06-10,IT
4,Diana,58000,2018-11-01,Marketing
5,Ethan,72000,2022-01-05,Operations
6,Fiona,55000,2017-05-25,HR
7,George,68000,2016-10-30,Finance
8,Hannah,72000,2023-04-12,IT
9,Ian,59000,2019-08-18,Marketing
10,Jane,75000,2021-02-20,Operations

```
   id    name salary start_date department
1   1   Alice  50000 15-03-2020         HR
2   2     Bob  60000 20-07-2019    Finance
3   3 Charlie  75000 10-06-2021         IT
4   4   Diana  58000 01-11-2018  Marketing
5   5   Ethan  72000 05-01-2022 Operations
6   6   Fiona  55000 25-05-2017         HR
7   7  George  68000 30-10-2016    Finance
8   8  Hannah  72000 12-04-2023         IT
9   9     Ian  59000 18-08-2019  Marketing
10 10    Jane  75000 20-02-2021 Operations
```

4) Create and Read CSV file

id,name,salary,start_date,department
1,Alice,50000,2020-03-15,HR
2,Bob,60000,2019-07-20,Finance
3,Charlie,75000,2021-06-10,IT
4,Diana,58000,2018-11-01,Marketing
5,Ethan,72000,2022-01-05,Operations
6,Fiona,55000,2017-05-25,HR
7,George,68000,2016-10-30,Finance

```
   id    name salary start_date department
1   1   Alice  50000 15-03-2020         HR
2   2     Bob  60000 20-07-2019    Finance
3   3 Charlie  75000 10-06-2021         IT
4   4   Diana  58000 01-11-2018  Marketing
5   5   Ethan  72000 05-01-2022 Operations
6   6   Fiona  55000 25-05-2017         HR
7   7  George  68000 30-10-2016    Finance
8   8  Hannah  72000 12-04-2023         IT
9   9     Ian  59000 18-08-2019  Marketing
10 10    Jane  75000 20-02-2021 Operations
```

8,Hannah,72000,2023-04-12,IT 9,Ian,59000,2019-08-18,Marketing 10,Jane,75000,2021-02-20,Operations	
5) Find maximum salary salary = max(data$salary) cat("Maximum salary is: ", salary, "\n")	Maximum salary is: 75000
6) Find the employee with maximum salary x=subset(data, salary==max(data$salary)) cat("Employee with maximum salary is: ", x$name, "\n")	Employee with maximum salary is: Charlie Jane
7) Find who is working in IT department x=subset(data, department=="IT") cat("Employees working in IT department are: ", x$name, "\n")	Employees working in IT department are: Charlie Hannah
8) How many employees have less than 60000 salary in the Marketing department x=subset(data, department=="Marketing" & salary<60000) cat("No of employees with less than 60000 salary in Marketing department are: ", nrow(x), "\n")	No of employees with less than 60000 salary in Marketing department are: 2
9) Employees joined after 2020 x=subset(data, as.Date(start_date, format="%y-%m-%d") > as.Date("2020-01-01")) cat("Employees joined after 2020 are: ", x$name, "\n")	Employees joined after 2020 are: Bob Fiona George Jane
10) Create and Read XLSX file i) Save as xlsx file ii) Read file library(readxl) data1 = read_excel("pr3_2.xlsx", sheet = 1) print(data1)	# A tibble: 10 x 5 id name salary start_date department <dbl> <chr> <dbl> <chr> <chr> 1 1 Alice 50000 2020-03-15 HR 2 2 Bob 60000 2019-07-20 Finance 3 3 Charlie 75000 2021-06-10 IT 4 4 Diana 58000 2018-11-01 Marketing 5 5 Ethan 72000 2022-01-05 Operations 6 6 Fiona 55000 2017-05-25 HR 7 7 George 68000 2016-10-30 Finance 8 8 Hannah 72000 2023-04-12 IT 9 9 Ian 59000 2019-08-18 Marketing 10 10 Jane 75000 2021-02-20 Operations

11) Find the employee with maximum salary x=subset(data, salary==max(data$salary)) cat("Employee with maximum salary is: ", x$name, "\n")	Employee with maximum salary is: Charlie Jane
12) Find who is working in IT department x=subset(data, department=="IT") cat("Employees working in IT department are: ", x$name, "\n")	Employees working in IT department are: Charlie Hannah
13) Employees joined after 2020 x=subset(data, as.Date(start_date, format="%y-%m-%d") > as.Date("2020-01-01")) cat("Employees joined after 2020 are: ", x$name, "\n")	Employees joined after 2020 are: Bob Fiona George Jane

Date – 03/02/2025

Practical No : 4

AIM – To study different types of analysis in Dataset

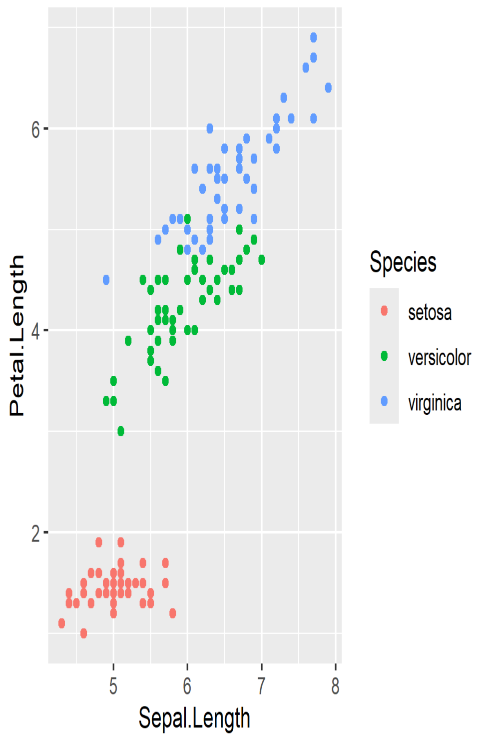
1) Install ggplot2 and read Iris dataset

```
install.packages("ggplot2")
#Iris
print(head(iris))
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	S
1	5.1	3.5	1.4	0.2	
2	4.9	3.0	1.4	0.2	
3	4.7	3.2	1.3	0.2	
4	4.6	3.1	1.5	0.2	
5	5.0	3.6	1.4	0.2	
6	5.4	3.9	1.7	0.4	

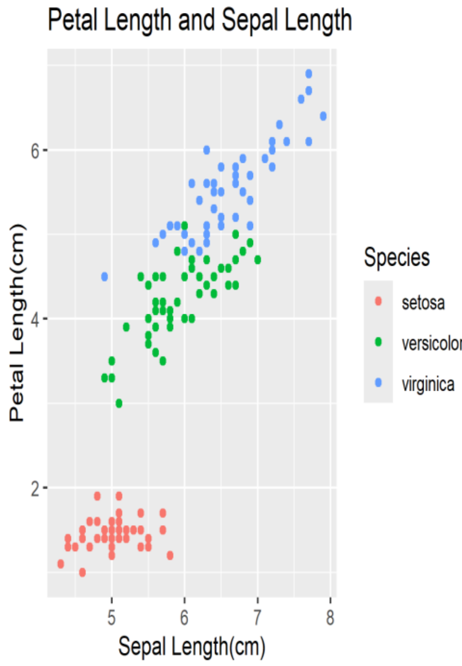
2) Plot Iris

```
library(ggplot2)
p = ggplot(iris, aes(Sepal.Length, Petal.Length,
color = Species)) + geom_point()
print(p)
```



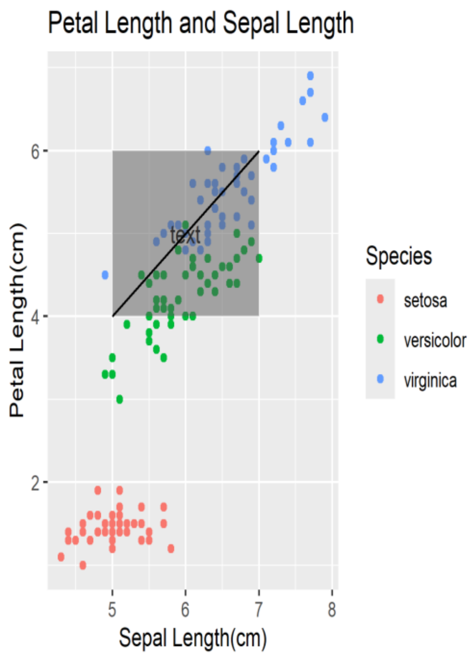
3) Plot iris x label and y label

```
p = ggplot(iris, aes(Sepal.Length, Petal.Length,
color = Species)) + geom_point() +labs(y="Petal
Length(cm)", x="Sepal
Length(cm)")+ggtitle("Petal Length and Sepal
Length")
```



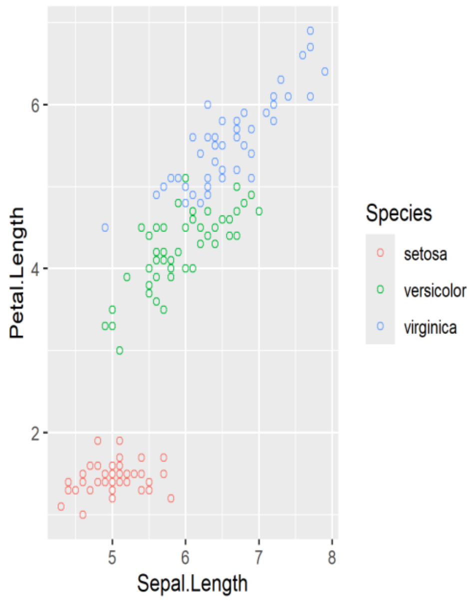
4) Iris plot with text, highlight area and segment

p = ggplot(iris, aes(Sepal.Length, Petal.Length, color = Species)) + geom_point() +labs(y="Petal Length(cm)", x="Sepal Length(cm)")+ggtitle("Petal Length and Sepal Length")+ annotate("text",x=6,y=5,label="text")+annotate("rect",xmin = 5, xmax=7, ymin=4, ymax=6, alpha=.5)+annotate("segment", x=5, xend=7, y=4, yend=6, color="black")



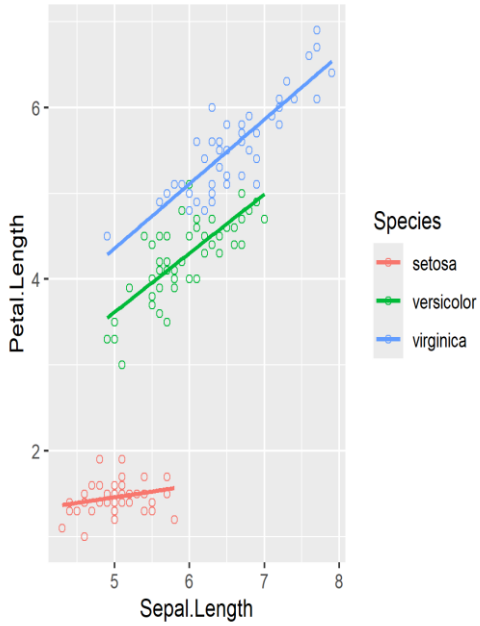
5) Change the shape of points

p=ggplot(iris, aes(Sepal.Length, Petal.Length, colour=Species)) + geom_point(shape=1)



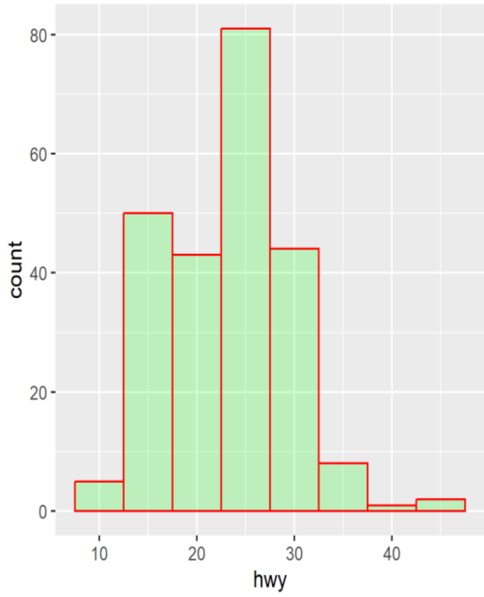
6) Add a line of best fit

p=ggplot(iris, aes(Sepal.Length, Petal.Length, colour=Species)) + geom_point(shape=1)+geom_smooth(method=lm)



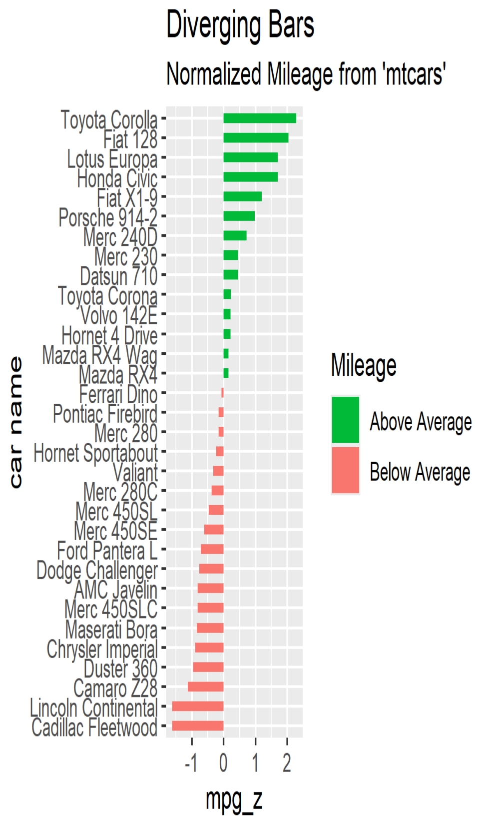
7) A bar count plot with a fill color

```
p=ggplot(data=mpg,
aes(x=hwy))+geom_histogram(col="red",
fill="green", alpha=.2,binwidth=5)
```

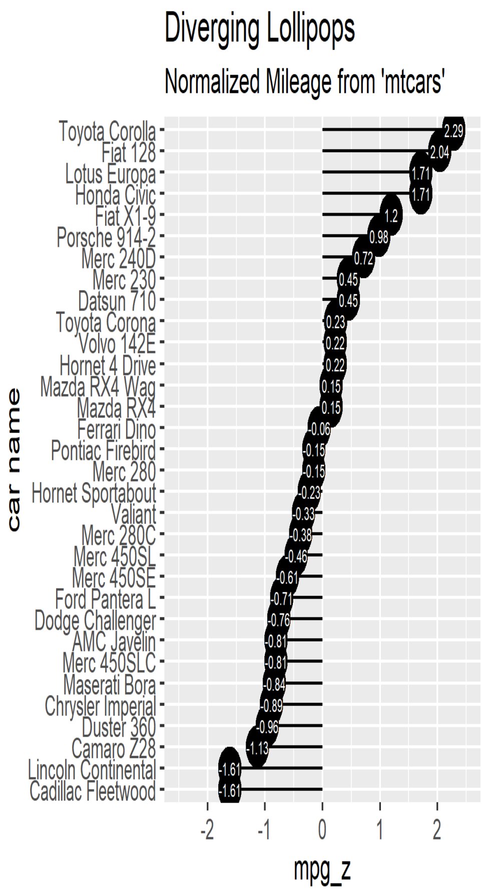


8) Add new coloum, sort it and print Diverging Barcharts

```
#Create new coloum for car names
mtcars$'car name' = rownames(mtcars)

#Compute normalized mpg
mtcars$mpg_z = round((mtcars$mpg -
mean(mtcars$mpg))/sd(mtcars$mpg),2)

#above/below avg flag
mtcars$mpg_type = ifelse(mtcars$mpg_z < 0,
"Below Average", "Above Average")

#sort
mtcars = mtcars[order(mtcars$mpg_z),]

#Convert to factor to retain sorted order in plot.
mtcars$'car name' = factor(mtcars$'car name',
levels=mtcars$'car name')

#Diverging Barcharts
p=ggplot(mtcars, aes(x=`car name`, y=mpg_z,
fill=mpg_type)) +
  geom_bar(stat="identity", width=0.5) +
  scale_fill_manual(name="Mileage",
labels=c("Above Average","Below Average"),
values=c("Above Average"="#00ba38", "Below
Average"="#f8766d")) +
  labs(subtitle="Normalized Mileage from
'mtcars'", title="Diverging Bars") +
  coord_flip()
```



9) Diverging Lollipop Chart




```
p=ggplot(mtcars, aes(x=`car name`,
y=mpg_z,label=mpg_z)) +
  geom_point(stat="identity", size=6, fill="black")
+ geom_segment(aes(y=0, xend=`car name`,
yend=mpg_z), color="black") +
geom_text(color="white", size=2) +
labs(title="Diverging Lollipops",
subtitle="Normalized Mileage from 'mtcars'")+
ylim(-2.5,2.5) + coord_flip()
```

10) Save plot and exec shell

```
# Save Plot
ggsave("plot.png", p, width=4, height=4,
units="in", dpi=300)
# Open Plot (Windows)
shell.exec("plot.png")
```

Date – 10/02/2025

Practical No : 5

AIM – To study linear regression in R and visualize the results with a scatter plot

1) Create dataframe and scatter plot

```
library(ggplot2)
data <- data.frame(
  Years_Exp = c(1.1, 1.3, 1.5, 2.0, 2.2,
2.9,3.0,3.2,3.2,3.7),
  Salary = c(39343, 46205, 37731, 43525,
39891,56642,60150,54445,64445,57189)
)

png("plot.png")
plot(data$Years_Exp, data$Salary,
     xlab = "Years Experienced",
     ylab = "Salary",
     main = "Scatter Plot of Years Experienced vs
Salary"
)
 dev.off()
```

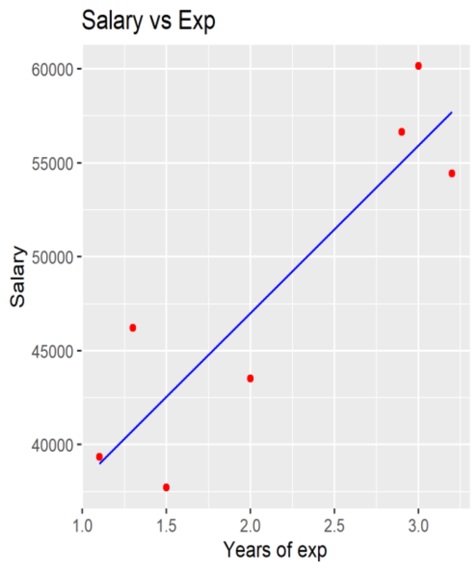
2) caTools installation and summary

```
install.packages('caTools')
library(caTools)
split = sample.split(data$Salary, SplitRatio = 0.7)
trainingset = subset(data, split == TRUE)
testset = subset(data, split == FALSE)
lm.r = lm(formula = Salary ~ Years_Exp, data =
trainingset)
summary(lm.r)
```

```
Call:
lm(formula = Salary ~ Years_Exp, data = trainingset)

Residuals:
    2     3     5     6     7     9    10
 6174 -4186 -8626  1524  4089  6498 -5473

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    27772       8077   3.438   0.0185 *
Years_Exp       9430       3018   3.124   0.0261 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6652 on 5 degrees of freedom
Multiple R-squared:  0.6612,    Adjusted R-squared:  0.5935
F-statistic: 9.759 on 1 and 5 DF,  p-value: 0.02614
```

3) New Dataset

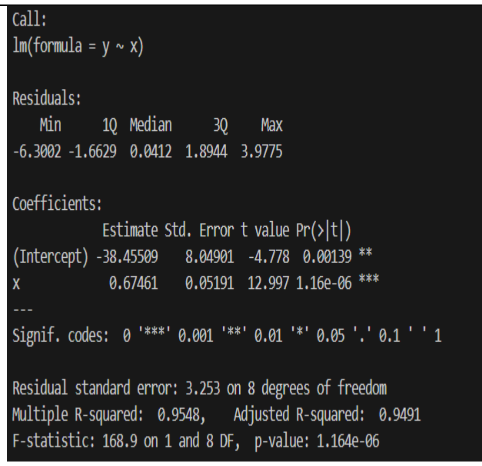
```
new_data = data.frame(Years_Exp = c(4.0,4.5,5.0))
predicted_sal = predict(lm.r, newdata = new_data)
p=ggplot()+geom_point(aes(x=trainingset$Years_Ex, y=trainingset$Salary), color='red')+geom_line(aes(x=trainingset$Years_Ex,y=predict(lm.r,newdata=trainingset)),color='blue')+ggtitle('Salary vs Exp')+xlab('Years of exp')+ylab('Salary')
 ggsave("plot.png", p, width=4,
 height=4,units="in", dpi=300)
shell.exec("plot.png")
```



Example 2

4) Summary

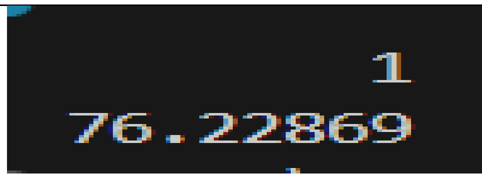
```
x=c(151,174,138,186,128,136,179,163,152,131)
y=c(63,81,56,91,47,57,76,72,62,48)
releation = lm(y~x)
print(summary(releation))
```



```
Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max 
-6.3002 -1.6629  0.0412  1.8944  3.9775 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -38.45509    8.04901  -4.778  0.00139 ** 
x             0.67461    0.05191  12.997 1.16e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.253 on 8 degrees of freedom
Multiple R-squared:  0.9548,	Adjusted R-squared:  0.9491 
F-statistic: 168.9 on 1 and 8 DF,  p-value: 1.164e-06
```

5) Predict the weight of new Persons

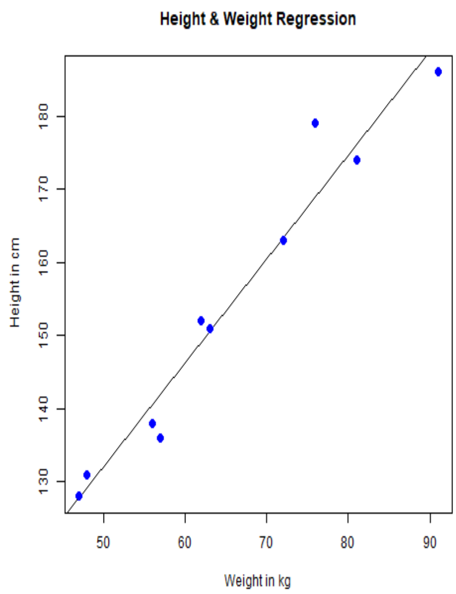
```
a=data.frame(x=170)
result=predict(releation,a)
print(result)
```



```
       1 
76.22869 
```

6) Plot the chart

```
png("plot.png")
plot(y,x,col="blue",main="Height & Weight Regression", abline(lm(x~y)), cex = 1.3, pch = 16, xlab = "Weight in kg", ylab = "Height in cm")
dev.off()
shell.exec("plot.png")
```

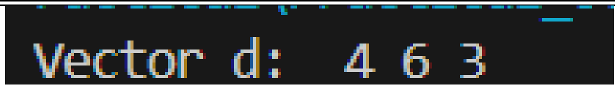
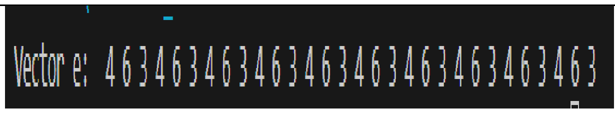
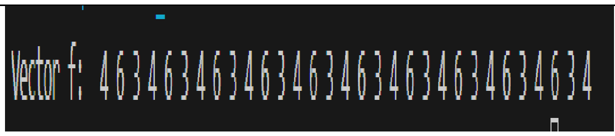
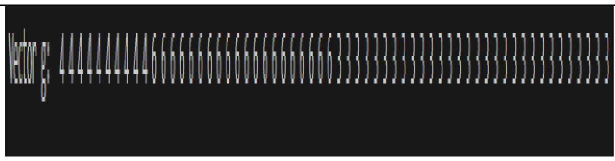
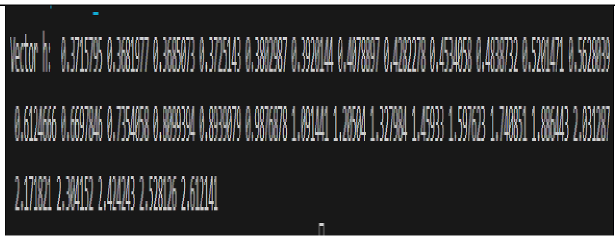


Date – 17/02/2025

Practical No : 6

AIM – Introduction to basic structure of R programming (variables assignment, data types, vector)

1) Variables and Assignment # 1. Assign the value of 44 to x x=44 # 2. Assign the value of 20 to y y=20 # 3. Make z the value of x – y, and display z. z = x - y cat("Value of z is: ", z, "\n")	3) Value of z is: 24
# 4. Calculate the square root of 2345, and perform a log2 transformation on the result. sqrval = sqrt(2345) log2sqr = log2(sqrval) cat("Log2 of the square root of 2345 is: ", log2sqr, "\n")	4) Log2 of the square root of 2345 is: 5.597686
# 5. Calculate the 10-based logarithm of 100, and multiply the result with the cosine of π. log10val = log10(100) cosval = cos(pi) result = log10val * cosval cat("Result of the multiplication of log10 of 100 and cosine of pi is: ", result, "\n")	5) Result of the multiplication of log10 of 100 and cosine of pi is: -2
2) Introduction to Vector # 1. Create the vectors # (a) (1; 2; 3; : : : ; 19; 20) vec_a = 1:20 cat("Vector a: ", vec_a, "\n")	Vector a: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
# (b) (20; 19; : : : ; 2; 1) vec_b = 20:1 cat("Vector b: ", vec_b, "\n")	Vector b: 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
# (c) (1; 2; 3; : : ; 19; 20; 19; 18; : ; 2; 1) vec_c = c(1:20, 19:1) cat("Vector c: ", vec_c, "\n")	Vector c: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

# (d) (4; 6; 3) and assign it to the name tmp. tmp = c(4, 6, 3) cat("Vector d: ", tmp, "\n")	
# (e) (4; 6; 3; 4; 6; 3; : : : ; 4; 6; 3) where there are 10 occurrences of 4. vec_e = rep(c(4, 6, 3), times = 10) cat("Vector e: ", vec_e, "\n")	
# (f) (4; 6; 3; 4; 6; 3; : : : ; 4; 6; 3; 4) where there are 11 occurrences of 4, 10 occurrences of 6 and 10 occurrences of 3. vec_f = c(rep(c(4, 6, 3), times = 10), 4) cat("Vector f: ", vec_f, "\n")	
# (g) (4; 4; : : : ; 4; 6; 6; : : : ; 6; 3; 3; : : : ; 3) where there are 10 occurrences of 4, 20 occurrences of 6 and 30 occurrences of 3. vec_g = c(rep(4, times = 10), rep(6, times = 20), rep(3, times = 30)) cat("Vector g: ", vec_g, "\n")	
# 2. Create a vector of the values of ex cos(x) at x = 3; 3:1; 3:2; : : : ; 6. x = seq(3, 6, by = 0.1) vec_h = exp(cos(x)) cat("Vector h: ", vec_h, "\n")	

Date – 24/02/2025

Practical No : 7

AIM – Logistic regression and Decision tree in R

1) Logistic Regression

data(mtcars)
print(head(mtcars))

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

input <- mtcars[,c("am","cyl","hp","wt")]
print(head(input))

	am	cyl	hp	wt
Mazda RX4	1	6	110	2.620
Mazda RX4 Wag	1	6	110	2.875
Datsun 710	1	4	93	2.320
Hornet 4 Drive	0	6	110	3.215
Hornet Sportabout	0	8	175	3.440
Valiant	0	6	105	3.460

am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)
print(summary(am.data))

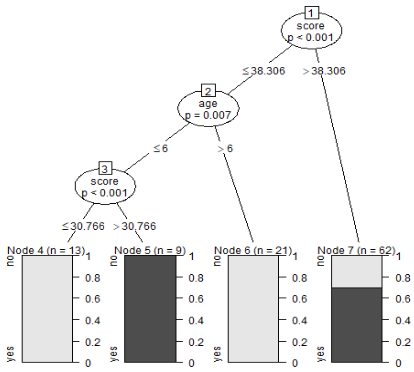
```
Call:
glm(formula = am ~ cyl + hp + wt, family = binomial, data = input)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 19.70288    8.11637   2.428   0.0152 *
cyl          0.48760    1.07162   0.455   0.6491
hp           0.03259    0.01886   1.728   0.0840 .
wt          -9.14947    4.15332  -2.203   0.0276 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.2297  on 31  degrees of freedom
Residual deviance:  9.8415  on 28  degrees of freedom
AIC: 17.841

Number of Fisher Scoring iterations: 8
```

2) Decision Tree

install.packages('party')
library(party)
input.dat = readingSkills[c(1:105),]
print(head(input.dat))

	nativeSpeaker	age	shoeSize	score
1	yes	5	24.83189	32.29385
2	yes	6	25.95238	36.63105
3	no	11	30.42170	49.60593
4	yes	7	28.66450	40.28456
5	yes	11	31.88207	55.46085
6	yes	10	30.07843	52.83124

```
png(file = "decision_tree.png")
output.tree <- ctree(
  nativeSpeaker ~ age + shoeSize + score,
  data = input.dat
)
plot(output.tree)
dev.off()
```



Date – 03/03/2025

Practical No : 8

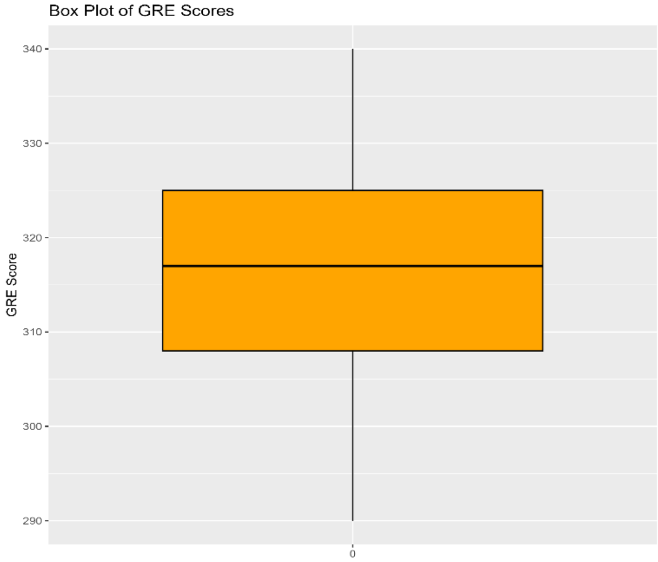
AIM – Admission Prediction Analysis using Linear Regression and Decision Tree in R

```
# Libraries
library(ggplot2)
library(dplyr)
library(caret)
library(rpart)
library(rpart.plot)
# Dataset load karo
data <- read.csv("Admission_Predict.csv")
# Columns ko rename karo simplicity ke liye
data <- data %>% rename(
  GRE = GRE.Score,
  TOEFL = TOEFL.Score,
  UnivRating = University.Rating,
  SOP = SOP,
  LOR = LOR,
  CGPA = CGPA,
  Research = Research,
  AdmitChance = Chance.of.Admit
)
# Missing values check karo
sum(is.na(data))
# Visualizations
# GRE Scores ka histogram
hist_plot <- ggplot(data, aes(x = GRE)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "black") +
  labs(title = "Histogram of GRE Scores", x = "GRE Score", y = "Frequency")
ggsave("Prac 8 plots/histogram_gre_scores.png", plot = hist_plot)
# GRE Score vs Chance of Admit scatter plot
scatter_plot <- ggplot(data, aes(x = GRE, y = AdmitChance)) +
  geom_point() +
  geom_smooth(method = "lm", col = "red") +
  labs(title = "GRE Score vs Chance of Admit", x = "GRE Score", y = "Chance of
Admit")
ggsave("Prac 8 plots/scatter_gre_vs_admit.png", plot = scatter_plot)
# GRE Score ka box plot
box_plot <- ggplot(data, aes(x = factor(0), y = GRE)) +
  geom_boxplot(fill = "orange", color = "black") +
  labs(title = "Box Plot of GRE Scores", x = "", y = "GRE Score")
ggsave("Prac 8 plots/box_plot_gre_scores.png", plot = box_plot)
```

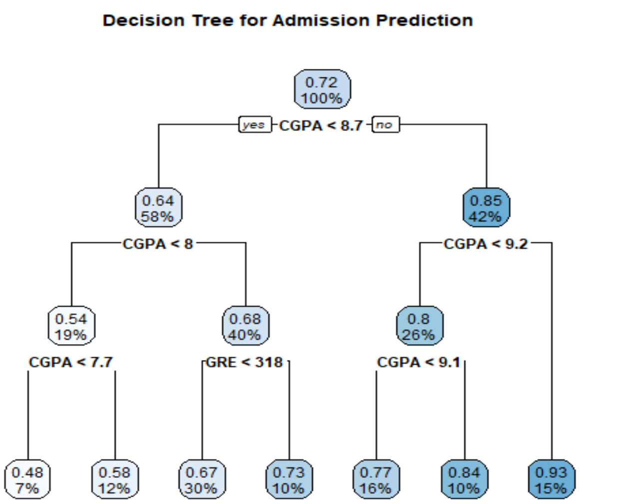
```
# Linear Regression
linear_model <- lm(AdmitChance ~ GRE + TOEFL + UnivRating + SOP + LOR +
CGPA + Research, data = data)
# Model ko predict karo
predictions <- predict(linear_model, newdata = data)
# Pair plot
png("Prac 8 plots/pair_plot.png")
pairs(data[, 2:9], main = "Pair Plot of Admission Predictors")
dev.off()
# Decision Tree
png("Prac 8 plots/decision_tree.png")
decision_tree <- rpart(AdmitChance ~ GRE + TOEFL + UnivRating + SOP + LOR +
CGPA + Research, data = data, method = "anova")
rpart.plot(decision_tree, main = "Decision Tree for Admission Prediction")
dev.off()
```

Output

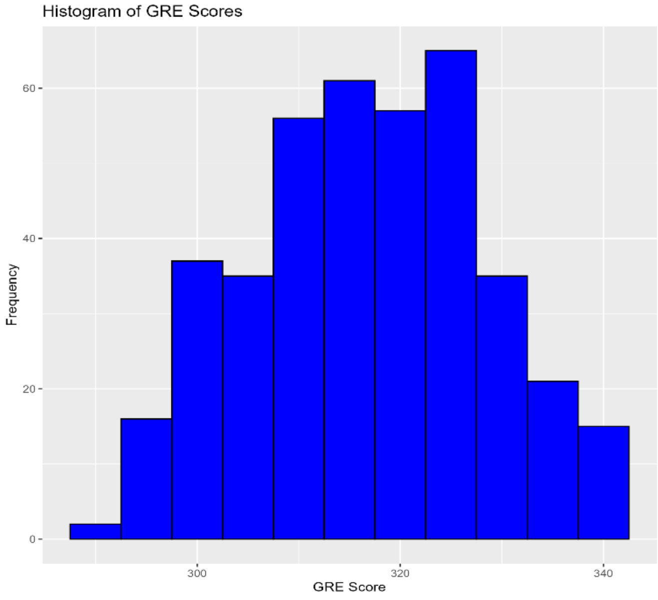
1) Box Plot



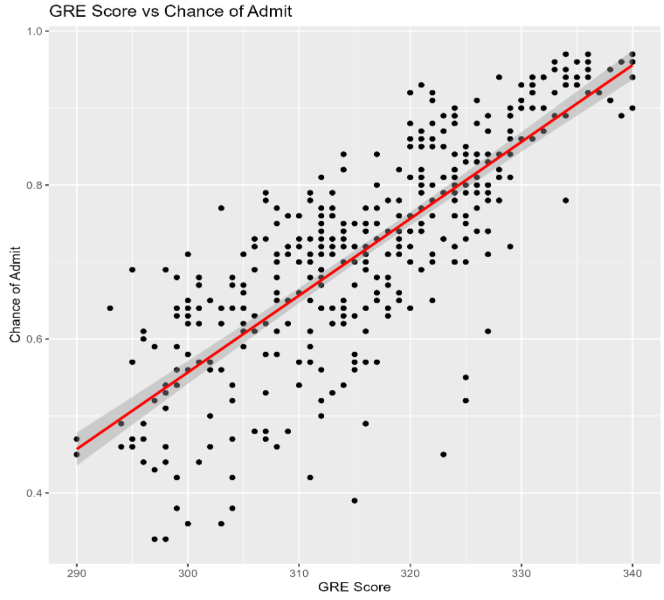
2) Decision Tree



3) Histogram



4) Scatter Plot



Date – 10/03/2025

Practical No : 9

AIM – R programming tasks for data manipulation and analysis

1) Create a numeric vector containing numbers 10 20 30 40 50 display the vector pr9 <- c(10, 20, 30, 40, 50) cat("Vector: ", pr9, "\n")	Vector: 10 20 30 40 50
2) Find the sum, mean and length of the vector cat("Sum: ", sum(pr9), "\n") cat("Mean: ", mean(pr9), "\n") cat("Length: ", length(pr9), "\n")	Sum: 150 Mean: 30 Length: 5
3) Extract 2 and 4th element from the vector cat("2nd Element: ", pr9[2], "\n") cat("4th Element: ", pr9[4], "\n")	2nd Element: 20 4th Element: 40
4) Create a 3x3 matrix with numbers from 1 to 9 and display it pr9 <- matrix(1:9, nrow = 3, ncol = 3) cat("Matrix: \n") print(pr9)	Matrix: [,1] [,2] [,3] [1,] 1 4 7 [2,] 2 5 8 [3,] 3 6 9
5) Extract the element from the second row and third coloumn cat("Element: ", pr9[2, 3], "\n")	Element: 8
6) Find the row wise and coloumn wise sum of the matrix cat("Row wise sum: ", rowSums(pr9), "\n") cat("Coloumn wise sum: ", colSums(pr9), "\n")	Row wise sum: 12 15 18 Coloumn wise sum: 6 15 24

7) Create and Read a csv file and display first 3 rows
pr9 <- data.frame(
  Name = c("Amit", "Ravi", "Priya", "Anita", "Vijay", "Sunita", "Raj", "Kiran", "Pooja", "Arjun", "Amit", "Ravi"),
  Age = c(23, 25, NA, 24, 26, 23, 27, 22, 24, NA, 23, 25),	

Score = c(85, 78, 90, 88, 76, 95, 89, 77, 92, 81, 85, 78)) write.csv(pr9, "pr9.csv", row.names = FALSE)	
8) Select and display only the age coloumn of the csv file cat("Age: ", pr9$Age, "\n")	Age: 23 25 NA 24 26 23 27 22 24 NA 23 25
9) Filter the rows from csv file where score is greater than 80 cat("Score > 80: \n") print(pr9[pr9$Score > 80,])	Score > 80: Name Age Score 1 Amit 23 85 3 Priya NA 90 4 Anita 24 88 6 Sunita 23 95 7 Raj 27 89 9 Pooja 24 92 10 Arjun NA 81 11 Amit 23 85
10) Update CSV to add a new coloumn grade to csv file based on marks/score pr9$Grade <- ifelse(pr9$Score > 90, "A", ifelse(pr9$Score > 80, "B", ifelse(pr9$Score > 70, "C", "D"))) write.csv(pr9, "pr9.csv", row.names = FALSE) cat("Grade values : ", pr9$Grade)	Grade values : B C B B C A B C A B B C
11) Find the highest and lowest value in the numeric coloumn cat("Highest: ", max(pr9$Score), "\n") cat("Lowest: ", min(pr9$Score), "\n")	Highest: 95 Lowest: 76
12) Show duplicate values in the csv file and update the csv file to remove duplicates cat("Duplicates: \n") print(pr9[duplicated(pr9),]) pr9 <- pr9[!duplicated(pr9),] write.csv(pr9, "pr9.csv", row.names = FALSE) cat("No Duplicates: \n") print(pr9[duplicated(pr9),])	Duplicates: Name Age Score Grade 11 Amit 23 85 B 12 Ravi 25 78 C No Duplicates: [1] Name Age Score Grade <0 rows> (or 0-length row.names)
13) Replace missing values with the mean of the column pr9$Age[is.na(pr9$Age)] <- mean(pr9$Age, na.rm = TRUE) write.csv(pr9, "pr9.csv", row.names = FALSE) cat("Updated CSV: \n") print(pr9)	Updated CSV: Name Age Score Grade 1 Amit 23.00 85 B 2 Ravi 25.00 78 C 3 Priya 24.25 90 B 4 Anita 24.00 88 B 5 Vijay 26.00 76 C 6 Sunita 23.00 95 A 7 Raj 27.00 89 B 8 Kiran 22.00 77 C 9 Pooja 24.00 92 A 10 Arjun 24.25 81 B

Date – 17/03/2025

Practical No : 10

AIM – Demonstrate the use of operators, conditional statements, loops, and functions in R

1) Create Two vectors. Apply following operations:

```
v <- c(2, 3, 4, 5, 6)
t <- c(5, 6, 7, 8, 9)
```

a) Arithmetic

```
cat("Addition: ", v + t, "\n")
cat("Subtraction: ", v - t, "\n")
cat("Multiplication: ", v * t, "\n")
cat("Division: ", v / t, "\n")
cat("Modulus: ", v %% t, "\n")
```

```
Addition:  7 9 11 13 15
Subtraction:  -3 -3 -3 -3 -3
Multiplication:  10 18 28 40 54
Division:  0.4 0.5 0.5714286 0.625 0.6666667
Modulus:  2 3 4 5 6
```

b) Relational

```
cat("Greater than: ", v > t, "\n")
cat("Less than: ", v < t, "\n")
cat("Equal to: ", v == t, "\n")
cat("Not Equal to: ", v != t, "\n")
```

```
Greater than:  FALSE FALSE FALSE FALSE FALS
Less than:  TRUE TRUE TRUE TRUE TRUE
Equal to:  FALSE FALSE FALSE FALSE FALSE
Not Equal to:  TRUE TRUE TRUE TRUE TRUE
```

c) Logical

```
cat("AND: ", v & t, "\n")
cat("OR: ", v | t, "\n")
cat("NOT: ", !v, "\n")
```

```
AND:  TRUE TRUE TRUE TRUE TRUE
OR:  TRUE TRUE TRUE TRUE TRUE
NOT:  FALSE FALSE FALSE FALSE FALSE
```

2) Write code in R to find greatest of two numbers

```
a <- 10
b <- 20
greatest <- if (a > b)
   a else b
cat("The greatest number is: ", greatest, "\n")
```

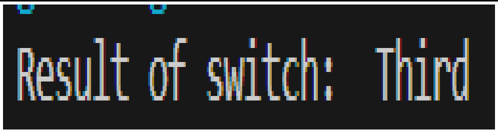
```
The greatest number is:  20
```

3) WAP to find factorial of a number

```
num <- 5
fact <- 1
for (i in 1:num)
   fact <- fact * i
cat("Factorial of ", num, " is: ", fact, "\n")
```

```
Factorial of  5  is:  120
```

4) WAP to demonstrate the use of switch in R

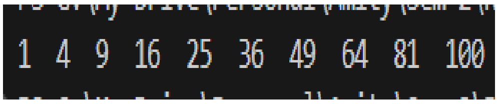



```
num <- 3
result <- switch(num,
  "one" = "First",
  "two" = "Second",
  "three" = "Third",
  "four" = "Fourth",
  "Invalid number"
)
cat("Result of switch: ", result, "\n")
```

5) WAP for fibonacci series using while loop in R


```
num <- 10
a <- 0
b <- 1
cnt <- 0
while (cnt < num) {
  cat(a, " ")
  c <- a + b
  a <- b
  b <- c
  cnt <- cnt + 1
}
cat("\n")
```

6) Create a function to print squares of numbers in sequence




```
squares <- function(n) {
  for (i in 1:n)
    cat(i^2, " ")
  cat("\n")
}
squares(10)
```

Time Series Analysis: Consider the annual rainfall details at a place starting from January 2012. We create an R time series object for a period of 12 months and plot it.

```
# Get the data points in form of a R vector.
rainfall <- c(799, 1174.8, 865.1, 1334.6, 635.4, 918.5, 685.5, 998.6, 784.2, 985, 882.8, 1071)

# Convert it to a time series object.
rainfall.timeseries <- ts(rainfall, start = c(2012, 1), frequency = 12)

# Print the timeseries data.
print(rainfall.timeseries)

# Load ggplot2 library
library(ggplot2)

# Create a data frame for plotting
rainfall_df <- data.frame(
   month = seq.Date(from = as.Date("2012-01-01"), by = "month", length.out = 12),
   rainfall = rainfall
)

# Plot the time series using ggplot2
p <- ggplot(rainfall_df, aes(x = month, y = rainfall)) +
   geom_line() +
   labs(title = "Monthly Rainfall", x = "Month", y = "Rainfall (mm)")

# Save the plot using ggsave
ggsave(filename = "g:/My Drive/Personal/Amity/Sem 2/R Programming/Codes/Practical/Prac 10 plots/rainfall.png", plot = p)
```