# R PROGRAM OUTPUTS

#1

name<-"ilakkiya"

age<-"19"

cat("Your name is", name, "\n")

cat("your age is", age, "\n")

cat("The version of R installation is:\n")

print(version)

#2

name = "Python";

n1 = 10;

n2=0.5

nums=c(10,20,30)

print(ls())

print("Details : ")

print(ls.str())

#3

num\_sequence<-20:50

mean\_seq<-mean(30:60)

sum\_seq<-sum(51:91)

cat("sequence of numbers",num\_sequence,"\n")

cat("sequence of mean",mean\_seq,"\n")

cat("sequence of sum",sum\_seq,"\n")

#4

v<-sample(-50:50,10,replace="TRUE")

cat("the contents of the vector:")

cat("10 random integers bet -50 to 50:")

cat(v)

#5

fib <- numeric(10)

fib[1] <- fib[2] <-1

for (i in 3:10) fib[i] <- fib[i-2]+fib[i-1]

print("First 10 Fibonacci Numbers : ")

print(fib)

#6

sieve\_of\_eratosthenes <- function(n) {

primes <- rep(TRUE, n+1)

primes[1] <- FALSE

for (i in 2:sqrt(n)) {

if (primes[i]) {

for (j in i^2:n) {

if (j %% i == 0) {

primes[j] <- FALSE

}

}

}

}

prime<-which(primes)

return(prime)

}

max\_number<-50

prime\_numbers<-sieve\_of\_eratosthenes(max\_number)

cat("Prime numbers up to", max\_number, ":", prime\_numbers, "\n")

#7

for (i in 1:100) {

if (i %% 3 == 0 && i %% 5 == 0) {

cat("FizzBuzz\n")

} else if (i %% 3 == 0) {

cat("Fizz\n")

} else if (i %% 5 == 0) {

cat("Buzz\n")

} else {

cat(i, "\n")

}

}

#8

letters <- c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j",

"k", "l", "m", "n", "o", "p", "q", "r", "s", "t",

"u", "v", "w", "x", "y", "z")

first\_10 <- letters[1:10]

last\_10 <- toupper(letters[17:26])

between\_22\_24 <- toupper(letters[22:24])

cat("The first 10 letters in lower case are:\n")

print(first\_10)

cat("The last 10 letters in upper case are:\n")

print(last\_10)

cat("The letters between 22nd to 24th letters in upper case are:\n")

print(between\_22\_24)

#9

print\_factors = function(n){

print(paste("The factors of", n,"are : "))

for(i in 1 : n){

if(n %% i == 0){

print(i)

}

}

}

print\_factors(10)

#10

v<-sample(1:10)

cat("the vector is :")

cat(v)

cat("the maximum value")

cat(max(v))

cat("the minimum value:")

cat(min(v))

#11

str = "List of Experiments"

cat("Original Vector (string) : ",str)

cat("\nUnique elements of the said vector : ",unique(str))

num=c(1,2,2,3,4,4,5,6)

cat("\nOriginal Vector (number) : ",num)

cat("\nUnique elements of the said vector : ",unique(num))

#12

a <- c(1,2,3)

b <- c(4,5,6)

c <- c(7,8,9)

m <- cbind(a,b,c)

cat("Content of Matrix : \n")

print(m)

#13

n=floor(rnorm(10,5,20))

print('List of Random numbers in the Normal Distribution')

print(n)

t=table(n)

print("Count Occurances of each Value")

print

#14

data=read.csv(file="C:/Statictics with R/input.csv",header=TRUE,sep=",")

print(data)

#15

a=c(3,2,6,7)

b=c("Red","Green","White","Blue")

c=c(TRUE,FALSE,TRUE,FALSE)

cat(a,"\nType : ",typeof(a),"\n")

cat(b,"\nType : ",typeof(b),"\n")

cat(c,"\nType : ",typeof(c))

#16

m1 = matrix(1:20, nrow=5, ncol=4)

print("5 × 4 matrix:")

print(m1)

m1 = matrix(1:9, nrow=3, ncol=3)

print("3 × 3 matrix:")

rnames = c("x", "y", "z")

cnames = c("a","b","c")

matrix = matrix(m1, nrow=3, ncol=3, byrow=TRUE, dimnames=list(rnames, cnames))

print(matrix)

m1 = matrix(c(6,7,8,2), nrow=2, ncol=2)

print("2 × 2 matrix:")

rnames = c("x", "y")

cnames = c("a","b")

matrix = matrix(m1, nrow=2, ncol=2, dimnames=list(rnames, cnames))

print(matrix)

#17

values <- c(1, 2, 3, 4, 5, 6)

dims <- c(2, 3)

my\_array <- array(values, dim = dims, dimnames = list(c("row1", "row2"), c("col1", "col2", "col3")))

my\_array

#18

v1=c(1,3,5,7)

v2=c(2,4,6,8,10)

a1=array(c(v1,v2),dim=c(3,3,2))

print(a1)

#19

l=list(c(1,2,2,5,7,12),month.abb,matrix(c(3,-8,1,-3),nrow=2),asin)

print("Content of List")

print(l)

#20

plot.new()

plot(1,type="n",xlab="",ylab="",xlim=c(0,10),ylim=c(0,10))

#21

print("Two Vectors of different Length : ")

v1=c(1,3,4,5)

v2=c(10,11,12,13,14,15)

print(v1)

print(v2)

result=array(c(v1,v2),dim=c(3,3,2))

print("New Array")

print(result)

cat("The Second row of the second matrix of the array : ",result[2,,2])

cat("\nThe element in the 3rd row and 3rd column of 1st matrix : ",result[3,3,1])

#22

a1 <- array(1:9, dim = c(3, 3))

a2 <- array(10:18, dim = c(3, 3))

a3 <- array(19:27, dim = c(3, 3))

print(a1)

print(a2)

print(a3)

a4 <- apply(cbind(a1, a2, a3), 1, function(x) rbind(x[1:3], x[4:6], x[7:9]))

print(a4)

#23

v <- 1:24

a <- array(v, dim = c(4, 3, 2))

print(a)

#24

a <- array(seq(from=50,length.out=15, by=2),c(5,3))

print("5X3 array of sequence > 50 : ")

print(a)

#25

exam\_data = data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin',

'Jonas'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')

)

subset\_exam\_data <- exam\_data[c(3, 5), c(1, 3)]

print(subset\_exam\_data)

#b

Country<-c("USA","USA","USA","USA","UK","USA","USA","India","USA","USA")

exam\_data1<- cbind(exam\_data, Country)

exam\_data1

#c

new\_exam\_data = data.frame(name = c('Robert','Sophia'),score = c(10.5,9),attempts = c('N','N'),qualify = c('yes','no'))

exam\_data <- rbind(exam\_data,new\_exam\_data)

print(exam\_data)

#d

sorted\_exam\_data <- exam\_data[order(exam\_data$name, exam\_data$score), ]

print(sorted\_exam\_data)

#e

write.table(exam\_data, file = "exam\_data.txt", row.names = FALSE)

file.info("exam\_data.txt")

#26

data(airquality)

is.data.frame(airquality)

airquality <- airquality[order(airquality$Month, airquality$Day), ]

airquality$Solar.R <- NULL

airquality$Wind <- NULL

print(airquality)

#27

data = women

print("Women data set of height and weight : ")

print(data)

height\_f=cut(women$height,3)

print("Factor coreesponding to height : ")

print(table(height\_f)

#28

L = sample(LETTERS,size=30,replace=TRUE)

print("Original data : ")

print(L)

f=factor(L)

print("Original Factors : ")

print(f)

print("Only five of the Letters : ")

print(table(L[1:5]))

#29

# Load the required library

library(datasets)

# Load the Iris dataset

data("iris")

# (i) Find dimension, Structure, Summary statistics, Standard Deviation of all features.

dim(iris) # Dimension of the dataset

str(iris) # Structure of the dataset

summary(iris) # Summary statistics

sapply(iris[,1:4],sd)

# (ii) Find mean and standard deviation of features grouped by three species of Iris flowers

aggregate(. ~ Species, data = iris, mean) # Mean by species

aggregate(. ~ Species, data = iris, sd) # Standard Deviation by species

# (iii) Find quantile value of sepal width and length

quantile(iris$Sepal.Width)

quantile(iris$Sepal.Length)

# (iv) Create new data frame named iris1 with a new column Sepal.Length.Cate that categorizes Sepal.Length by quantile

iris1 <- iris

iris1$Sepal.Length.Cate <- cut(iris$Sepal.Length, breaks = quantile(iris$Sepal.Length), labels = FALSE)

# (v) Average value of numerical variables by two categorical variables: Species and Sepal.Length.Cate

aggregate(. ~ Species + Sepal.Length.Cate, data = iris1, mean)

# (vi) Average mean value of numerical variables by Species and Sepal.Length.Cate

aggregate(. ~ Species + Sepal.Length.Cate, data = iris1, mean)

# (vii) Create Pivot Table based on Species and Sepal.Length.Cate

pivot\_table <- table(iris1$Species, iris1$Sepal.Length.Cate)

#30

data(iris)

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

y <- iris[, "Species"]

set.seed(42)

train\_index <- sample(1:nrow(iris), 0.8 \* nrow(iris))

X\_train <- X[train\_index, ]

X\_test <- X[-train\_index, ]

y\_train <- y[train\_index]

y\_test <- y[-train\_index]

model <- glm(y\_train ~ ., data = X\_train, family = "binomial")

y\_pred <- predict(model, newdata = X\_test, type = "response")

print(y\_pred)

cm <- table(y\_test, y\_pred > 0.5)

print(cm)

#31

v <- c(90,50,70,80,70,60,20,30,80,90,20)

cat("Mean : ",sum(v)/length(v))

cat("\nMedian : ",median(v))

mode <- function()

{

return(names(sort(-table(v)))[1])

}

mm <- mode()

cat("\nMode : ",mm)

v <- unique(v)

cat("\n2nd Highest value of the given vector : ",sort(v)[length(v)-1])

cat("\n3rd lowest value of the given vector : ",sort(v)[3])

#32

data(airquality)

sum\_temp <- sum(airquality$Temp, na.rm = TRUE)

n\_temp <- length(airquality$Temp[!is.na(airquality$Temp)])

mean\_temp <- sum\_temp / n\_temp

mean\_temp

# ii.

head(airquality, n = 5)

# iii.

install.packages("dplyr")

library(dplyr)

select(airquality, -Temp, -Wind)

# iv.

min\_temp <- min(airquality$Temp, na.rm = TRUE)

min\_temp

coldest\_day <- which(airquality$Temp == min\_temp)

coldest\_day

airquality[coldest\_day, ]

# v.

sum(airquality$Wind > 17, na.rm = TRUE)

#33

#i

data(airquality)

summary(airquality)

#ii

install.packages("reshape")

library(reshape)

melted <- melt(airquality)

head(melted)

#iii

melted <- melt(airquality, id = c("Month", "Day"))

head(melted)

#iv

casted <- cast(melted, Month + Day ~ variable)

head(casted)

#v

casted <- cast(melted, Month ~ variable, mean)

casted

#34

handle\_missing\_values <- function(data, threshold = 0.1){

num\_rows <- nrow(data)

missing\_counts <- colSums(is.na(data))

for(col in names(data)){

missing\_count <- missing\_counts[col]

if(missing\_count > 0){

if(missing\_count/num\_rows < threshold){

data <- data[complete.cases(data),]

} else {

mean\_val <- mean(data[[col]],na.rm=TRUE)

data[[col]][is.na(data[[col]])] <- mean\_val

}

}

}

return (data)

}

df\_handled <- handle\_missing\_values(airquality)

print(df\_handled)

# Apply linear regression algorithm using Least Squares method on "Ozone" and "Solar.R"

model <- lm(Ozone ~ Solar.R, data = airquality)

# Plot scatter plot between Ozone and Solar.R and add regression line created by above model

plot(airquality$Solar.R, airquality$Ozone)

abline(model,col="red")

#35

# Load necessary libraries

library(dplyr)

library(tidyr)

# Load the ChickWeight dataset if not already loaded

data("ChickWeight")

print(ChickWeight)

# (i) Order the data frame by "weight" in ascending order and extract the last 6 records for each diet

#ordered\_chick\_weight <- ChickWeight %>%

#arrange(Diet, weight) %>%

#group\_by(Diet) %>%

#tail(n = 6)

ordered\_data <- ChickWeight[order(ChickWeight$weight), ]

last\_six\_records <- tail(ordered\_data, 6)

print(last\_six\_records)

# (ii.a) Melting the data based on "Chick", "Time", and "Diet" as ID variables

melted\_data <- ordered\_chick\_weight %>%

pivot\_longer(cols = starts\_with("weight"), names\_to = "Time of Diet", values\_to = "Weight")

print(melted\_data)

# (ii.b) Casting to display the mean value of weight grouped by Diet

mean\_weight\_by\_diet <- melted\_data %>%

group\_by(Diet, Time) %>%

summarise(mean\_weight = mean(Weight)) %>%

pivot\_wider(names\_from = Time, values\_from = mean\_weight)

print(mean\_weight\_by\_diet)

# (ii.c) Casting to display the mode of weight grouped by Diet

mode\_weight\_by\_diet <- melted\_data %>%

group\_by(Diet, Time) %>%

summarise(mode\_weight = as.numeric(names(table(Weight)[which.max(table(Weight))]))) %>%

pivot\_wider(names\_from = Time, values\_from = mode\_weight)

print(mode\_weight\_by\_diet)

#36

# Load required libraries

library(ggplot2)

# Load the ChickWeight dataset

data(ChickWeight)

# (a) Create Box plot for "weight" grouped by "Diet"

boxplot\_plot <- ggplot(ChickWeight, aes(x = as.factor(Diet), y = weight)) +

geom\_boxplot() +

labs(x = "Diet", y = "Weight", title = "Box Plot of Weight Grouped by Diet")

print(boxplot\_plot)

# (b) Create Histogram for "weight" features belonging to Diet-1 category

histogram\_plot <- ggplot(ChickWeight, aes(x = weight)) +

geom\_histogram(binwidth = 5, fill = "blue", color = "black") +

labs(x = "Weight", y = "Frequency", title = "Histogram of Weight for Diet-1")

print(histogram\_plot)

# (c) Create Scatter plot for "weight" vs "Time" grouped by Diet

scatter\_plot <- ggplot(ChickWeight, aes(x = Time, y = weight, color = as.factor(Diet))) +

geom\_point() +

labs(x = "Time", y = "Weight", title = "Scatter Plot of Weight vs Time Grouped by Diet")

print(scatter\_plot)

#37

# Load the ChickWeight dataset if not already loaded

data("ChickWeight")

# Convert "Diet" to a factor

ChickWeight$Diet <- as.factor(ChickWeight$Diet)

# (a) Create multi regression model to predict weight using "Time" and "Diet"

multi\_reg\_model <- lm(weight ~ Time + Diet, data = ChickWeight)

# (b) Predict weight for Time=10 and Diet=1

new\_data <- data.frame(Time = 10, Diet = factor(1)) # Convert Diet to factor

predicted\_weight <- predict(multi\_reg\_model, newdata = new\_data)

cat("Predicted weight for Time = 10 and Diet = 1:", predicted\_weight, "\n")

# (c) Calculate the error in the model for the prediction

actual\_weight <- ChickWeight$weight[ChickWeight$Time == 10 & ChickWeight$Diet == 1]

error <- actual\_weight - predicted\_weight

cat("Error:", error)

#38

# Load necessary libraries

library(dplyr)

# Load the Titanic dataset if not already loaded

data("Titanic")

# Convert Titanic array to a data frame

titanic\_df <- as.data.frame(Titanic)

# (a) Draw a Bar chart to show details of "Survived" on the Titanic based on passenger Class

survival\_class <- titanic\_df %>%

group\_by(Class, Survived) %>%

summarise(Count = sum(Freq))

barplot(matrix(survival\_class$Count, ncol = 2), beside = TRUE,

main = "Survival Count by Passenger Class",

xlab = "Survived", ylab = "Count", col = c("blue", "red"),

legend = levels(survival\_class$Class))

# (b) Modify the above plot based on gender of people who survived

survival\_gender <- titanic\_df %>%

group\_by(Sex, Class, Survived) %>%

summarise(Count = sum(Freq))

par(mfrow = c(1, 2)) # Create side-by-side plots

for (gender in unique(survival\_gender$Sex)) {

subset\_data <- subset(survival\_gender, Sex == gender)

barplot(matrix(subset\_data$Count, ncol = 2), beside = TRUE,

main = paste("Survival Count by Passenger Class (", gender, ")", sep = ""),

xlab = "Survived", ylab = "Count", col = c("blue", "red"),

legend = levels(subset\_data$Class))

}

par(mfrow = c(1, 1)) # Reset the layout

# (c) Draw histogram plot to show distribution of feature "Age"

numeric\_age <- as.numeric(titanic\_df$Age)

hist(numeric\_age[!is.na(numeric\_age)], breaks = seq(0, 100, by = 5), col = "blue", main = "Distribution of Age",

xlab = "Age", ylab = "Frequency")

#39

#view structure of data

str(USArrests)

#view summary statistics of data

summary(USArrests)

data("USArrests")

#find the index of the maximum value in Rape column

max\_rape <- which.max(USArrests$Rape)

#print the state name corresponding to that index

rownames(USArrests)[max\_rape]

data("USArrests")

#find the index of the maximum value in Murder column

max\_murder <- which.max(USArrests$Murder)

#print the state name corresponding to that index

rownames(USArrests)[max\_murder]

#find the index of the minimum value in Murder column

min\_murder <- which.min(USArrests$Murder)

#print the state name corresponding to that index

rownames(USArrests)[min\_murder]

#40

# Create a data frame based on the table

df <- data.frame(Month = 1:12, Spends = c(1000, 4000, 5000, 4500, 3000, 4000, 9000, 11000, 15000, 12000, 7000, 3000), Sales = c(9914, 40487, 54324, 50044, 34719, 42551, 94871, 118914, 158484, 131348, 78504, 36284))

df

# Create a regression model for Sales based on Spends

model <- lm(Sales ~ Spends, data = df)

model

# Predict the Sales if Spend = 13500

predict(model, newdata = data.frame(Spends = 13500))