ITA0435 – STATISTICS WITH R PROGRAMMING FOR VISUALIZATION

NAME ; R.SABARISH

REG NO : 192324092

R LAB EXPERIMENTS

1.Create numeric, character, and logical vectors and display type and content.

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

a

print(paste("Type: ",class(a)))

b = c("Gokul","Raghul","Dhanush")

b

print(paste("Type: ",class(b)))

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

c

print(paste("Type : ",class(c)))

```
> a = c(1,2,3,4,5,6)
> a
[1] 1 2 3 4 5 6
> print(paste("Type: ",class(a)))
[1] "Type:  numeric"
> b = c("Gokul","Raghul","Dhanush")
> b
[1] "Gokul"   "Raghul"  "Dhanush"
> print(paste("Type: ",class(b)))
[1] "Type:  character"
> c = c(TRUE,FALSE,TRUE,FALSE)
> c
[1]  TRUE FALSE  TRUE FALSE
> print(paste("Type : ",class(c)))
[1] "Type :  logical"
> 
```

2.Create labeled matrices (5×4, 3×3, 2×2) filled by row/column.

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

a

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

b

c = matrix(1:4,nrow=2,ncol=2)

c

```
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
> b = matrix(1:9,nrow=3,ncol=3)
> b
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> c = matrix(1:4,nrow=2,ncol=2)
> c
```

3. Write an R program to create and display a 3D array with specified rows, columns, and tables.

a = 1.24

arr = array(a, dim=c(2,3,4))

arr

```
     [,1] [,2]
[1,]    1    3
[2,]    2    4
> a = 1.24
> arr = array(a, dim=c(2,3,4))
> arr
, , 1

     [,1] [,2] [,3]
[1,] 1.24 1.24 1.24
[2,] 1.24 1.24 1.24

, , 2

     [,1] [,2] [,3]
[1,] 1.24 1.24 1.24
[2,] 1.24 1.24 1.24

, , 3

     [,1] [,2] [,3]
[1,] 1.24 1.24 1.24
[2,] 1.24 1.24 1.24

, , 4
```

4. Create arrays from vectors with dimension names, print specific elements.

a = 1:8

b = array(a, dim=c(2,2,2))

b

print(b[2,1,2])

```
> a = 1:8
> b = array(a, dim=c(2,2,2))
> b
, , 1

     [,1] [,2]
[1,]    1    3
[2,]    2    4

, , 2

     [,1] [,2]
[1,]    5    7
[2,]    6    8

> print(b[2,1,2])
[1] 6
>
```

5. Create and manipulate factor variables (e.g., women's dataset heights, random LETTERS sample)

a = c("Short","Medium","Tall","Medium","Tall","Short")

b = factor(a)

print(b)

set.seed(10)

c= sample(LETTERS[1:5],8,replace = TRUE)

d = factor(c)

print(d)

```
> a = c("Short","Medium","Tall","Medium","Tall","Short")
> b = factor(a)
> print(b)
[1] Short  Medium Tall   Medium Tall   Short
Levels: Medium Short Tall
> set.seed(10)
> c= sample(LETTERS[1:5],8,replace = TRUE)
> d = factor(c)
> print(d)
[1] C A B D C B B B
Levels: A B C D
>
```

6. Create an R list containing vectors, matrices, and functions; display contents.

```
a = c(1,2,3,4)

b = matrix(1:6, nrow = 2,ncol =3)

c = function(x)x^2

mylist=list(Vector = a, Matrix = b , Function = c)

print(mylist)
```

```
$Vector
[1] 1 2 3 4

$Matrix
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

$Function
function(x)x^2

>
```

7. Write R programs for basic tasks: Factors of a number, generate a vector of 10 random integers between -50 and 50, print numbers 1–100 with FizzBuzz logic.

```
n<-12L

factors<- integer(0)

for(i in seq_len(n)){

 if(n %% i == 0L){

  factors <-c(factors,i)

 }

}

print("factors of the number:")

print(factors)

set.seed(123)

print("10 random integers:")

print(sample(-50:50,10,replace=TRUE))
```

```
for(i in 1:100){
 if(i %% 15 == 0) print("Fizzbuzz")
 else if (i %% 3 == 0) print("Fizz")
 else if (i %% 5 == 0) print("Buzz")
 else  print(i)
}
```

OUTPUT :

```
[1] 1
[1] 2
[1] "Fizz"
[1] 4
[1] "Buzz"
[1] "Fizz"
[1] 7
[1] 8
[1] "Fizz"
[1] "Buzz"
[1] 11
[1] "Fizz"
[1] 13
[1] 14
[1] "Fizzbuzz"
[1] 16
[1] 17
[1] "Fizz"
[1] 19
[1] "Buzz"
[1] "Fizz"
[1] 22
[1] 23
[1] "Fizz"
[1] "Buzz"
[1] 26
[1] "Fizz"
[1] 28
[1] 29
[1] "Fizzbuzz"
[1] 31
[1] 32
[1] "Fizz"
[1] 34
[1] "Buzz"
[1] "Fizz"
[1] 37
[1] 38
[1] "Fizz"
[1] "Buzz"
[1] 41
[1] "Fizz"
[1] 43
[1] 44
[1] "Fizzbuzz"
[1] 46
[1] 47
[1] "Fizz"
[1] 49
[1] "Buzz"
[1] "Fizz"
[1] 52
[1] 53
[1] "Fizz"
[1] "Buzz"
```

8. Generate random numbers from a normal distribution; count occurrences.

set.seed(123)

x <- rnorm(100)

y<-(table(round(x)))

print(as.data.frame(y))

OUTPUT :

```
  Var1 Freq
1   -2    4
2   -1   21
3    0   45
4    1   22
5    2    8
> 
```

9. Create empty plots with specified axis limits.

rm(c)

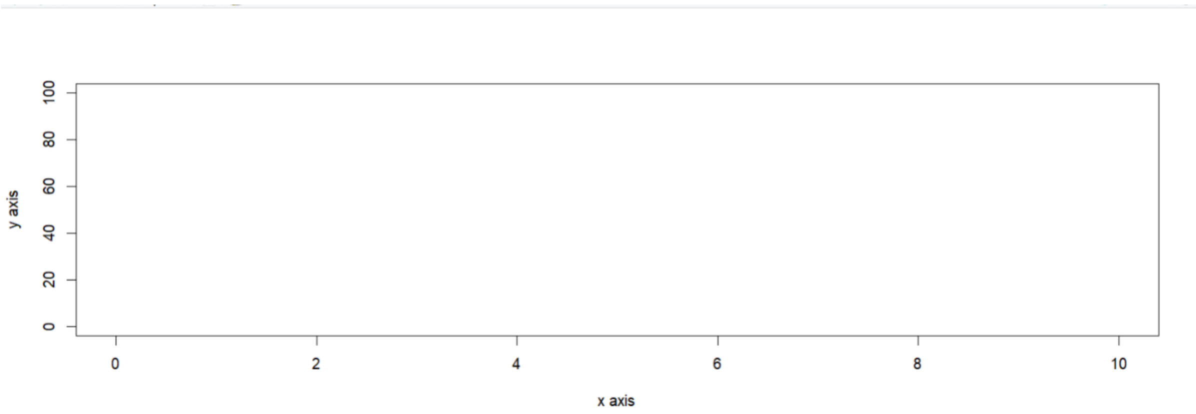
plot(1, type = "n",

  xlim = c(0,10),

  ylim = c(0,100),

  xlab = "x axis",

  ylab = "y axis")



10. Create and explore a data frame exam_data with name, score, attempts, and qualifying fields. Perform extract, add row/column, sort, save to file.

```
x <- data.frame(
  name = c("Anu","Bala","Charan","Divya","Esha"),
  score = c(85,62,90,75,88),
  attempts = c(1,3,2,2,1),
  qualify = c("yes","no","yes","no","yes")
)
x <-rbind(x, data.frame(name="Farhan",score=70,attempts=3,qualify="no"))
x$grade <- c("A","C","A+","B","A","B")
x <- x[order(-x$score),]
write.csv(x,"x.csv",row.names=FALSE)
print(x)
```

OUTPUT :

```
    name score attempts qualify grade
3 Charan    90        2     yes    A+
5   Esha    88        1     yes     A
1    Anu    85        1     yes     A
4  Divya    75        2      no     B
6 Farhan    70        3      no     B
2   Bala    62        3      no     C
> 
```

11. Write an R program to read a .csv file and display contents.

```
data <- read.csv("students_copy.csv")
print("Contents of the CSV file:")
print(data)
```

OUTPUT :

```
      Name.Marks.Age
1 1    Arun     85  20
2 2    Bala     78  21
3 3 Charan      92  19
4 4   Divya     88  20
5 5    Esha     74  22
> print("Contents of the CSV file:")
```

12. Perform data reshaping on air quality dataset: melt, cast, compute monthly averages for Ozone, Solar.R, Wind, and Temperature.

```
data("airquality")

result <- aggregate(cbind(Ozone, Solar.R, Wind, Temp) ~ Month,

          data = airquality,

          FUN = mean,

          na.rm = TRUE)

print("Monthly average values:")

print(result)
```

```
  Month    Ozone  Solar.R      Wind     Temp
1     5 24.12500 182.0417 11.504167 66.45833
2     6 29.44444 184.2222 12.177778 78.22222
3     7 59.11538 216.4231  8.523077 83.88462
4     8 60.00000 173.0870  8.860870 83.69565
5     9 31.44828 168.2069 10.075862 76.89655
> print("Contents of the CSV file:")
```

13. Combine multiple arrays row-wise

```
rm(list = ls())

a <- c(1, 2, 3)

b <- c(4, 5, 6)

c <- c(7, 8, 9)

d <- rbind(a, b, c)
```

print("Combined arrays (row-wise):")

print(d)

```
> print(d)
  [,1] [,2] [,3]
a    1    2    3
b    4    5    6
c    7    8    9
> print("Contents of the CSV file:")
```

14. Explore and manipulate ChickWeight dataset (sorting, melting, casting by Diet).

data("ChickWeight")

head(ChickWeight)

sorted <- ChickWeight[order(ChickWeight$weight),]

print(sorted)

aggregate(weight ~ Diet, ChickWeight, mean)

OUTPUT :

```
392      87    6     35     3
137      88   10     12     1
204      88   14     19     1
77       89    8      7     1
114      89   12     10     1
188      89   10     17     1
216      89   14     20     1
441      89    8     39     3
102      90   12      9     1
 [ reached 'max' / getOption("max.print") -- omitted 328 rows ]
> aggregate(weight ~ Diet, ChickWeight, mean)
  Diet   weight
1    1 102.6455
2    2 122.6167
3    3 142.9500
4    4 135.2627
```

15. Perform EDA on iris dataset: dimensions, summary, standard deviation,quantiles, grouping by Species, pivot table, categorical grouping with Sepal.Length categories.

data("iris")

dim(iris)

summary(iris)

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

quantile(iris$Sepal.Length)

aggregate(. ~ Species, iris, mean)

OUTPUT :

```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa    :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
> sapply(iris[,1:4], sd)
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   0.8280661    0.4358663    1.7652982    0.7622377
> quantile(iris$Sepal.Length)
  0%  25%  50%  75% 100%
 4.3  5.1  5.8  6.4  7.9
> aggregate(. ~ Species, iris, mean)
     Species Sepal.Length Sepal.Width Petal.Length Petal.Width
1     setosa        5.006       3.428        1.462       0.246
2 versicolor        5.936       2.770        4.260       1.326
3  virginica        6.588       2.974        5.552       2.026
> print("Contents of the CSV file:")
```

16. Explore USArrests dataset: summary statistics, state with largest rape arrests, max & min murder rates, correlation among features, states above median assault arrests and bottom 25% for murder, visualization with histogram, density, scatterplots, bar graphs.

data("USArrests")

summary(USArrests)

which.max(USArrests$Rape)

max(USArrests$Murder)

min(USArrests$Murder)

cor(USArrests)

hist(USArrests$Murder)

```
> data("USArrests")
> summary(USArrests)
     Murder          Assault         UrbanPop          Rape      
 Min.   : 0.800   Min.   : 45.0   Min.   :32.00   Min.   : 7.30  
 1st Qu.: 4.075   1st Qu.:109.0   1st Qu.:54.50   1st Qu.:15.07  
 Median : 7.250   Median :159.0   Median :66.00   Median :20.10  
 Mean   : 7.788   Mean   :170.8   Mean   :65.54   Mean   :21.23  
 3rd Qu.:11.250   3rd Qu.:249.0   3rd Qu.:77.75   3rd Qu.:26.18  
 Max.   :17.400   Max.   :337.0   Max.   :91.00   Max.   :46.00  
> which.max(USArrests$Rape)
[1] 28
> max(USArrests$Murder)
[1] 17.4
> min(USArrests$Murder)
[1] 0.8
> cor(USArrests)
             Murder   Assault   UrbanPop      Rape
Murder   1.00000000 0.8018733 0.06957262 0.5635788
Assault  0.80187331 1.0000000 0.25887170 0.6652412
UrbanPop 0.06957262 0.2588717 1.00000000 0.4113412
Rape     0.56357883 0.6652412 0.41134124 1.0000000
> hist(USArrests$Murder)
> print("Contents of the CSV file:")
```

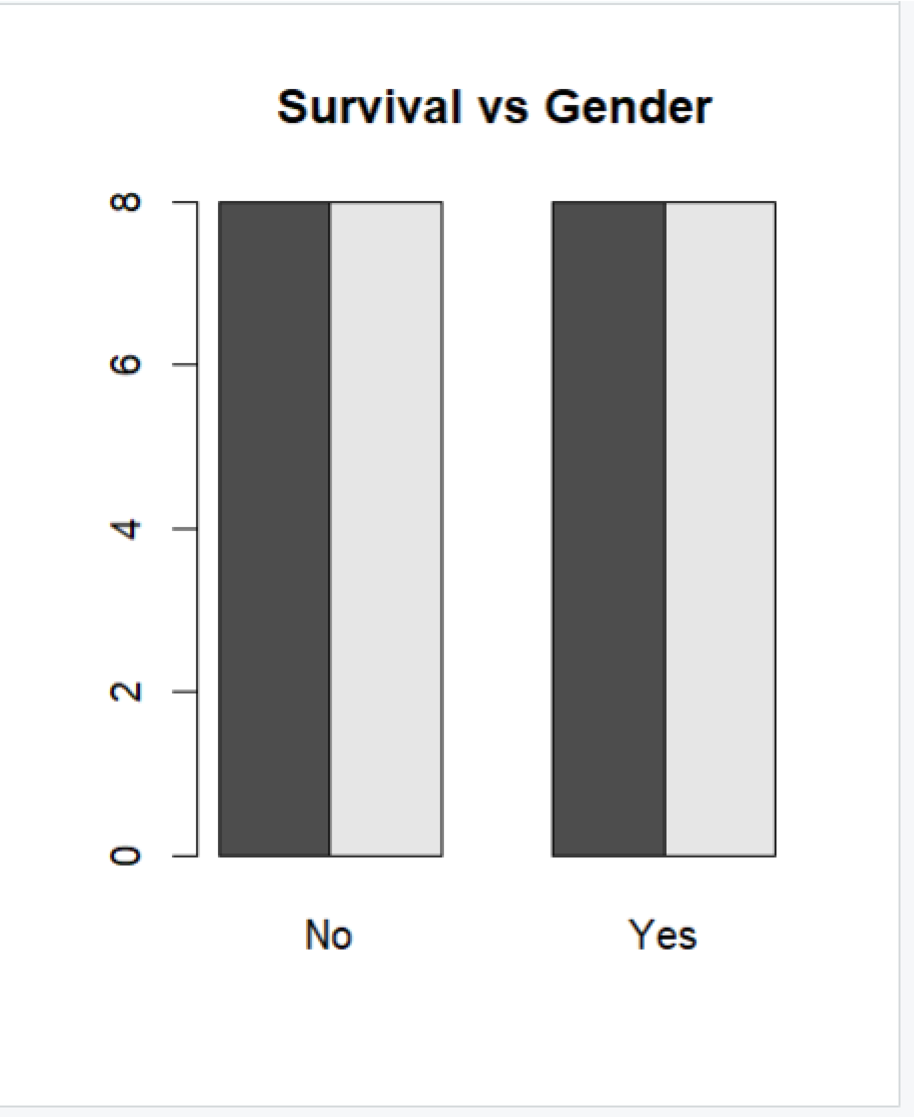
17. Explore Titanic dataset: bar chart of survival vs class, modify plot by gender, histogram of Age.

```
data("Titanic")

t <- as.data.frame(Titanic)

barplot(table(t$Class, t$Survived), main="Survival vs Class", beside=TRUE)

barplot(table(t$Sex, t$Survived), main="Survival vs Gender", beside=TRUE)
```



18. Create graphs in R: boxplot, histogram, bar plot, line chart, scatter plot.

x <- c(10, 20, 30, 40, 50)

y <- c(5, 15, 25, 35, 45)

par(mfrow = c(2, 3))

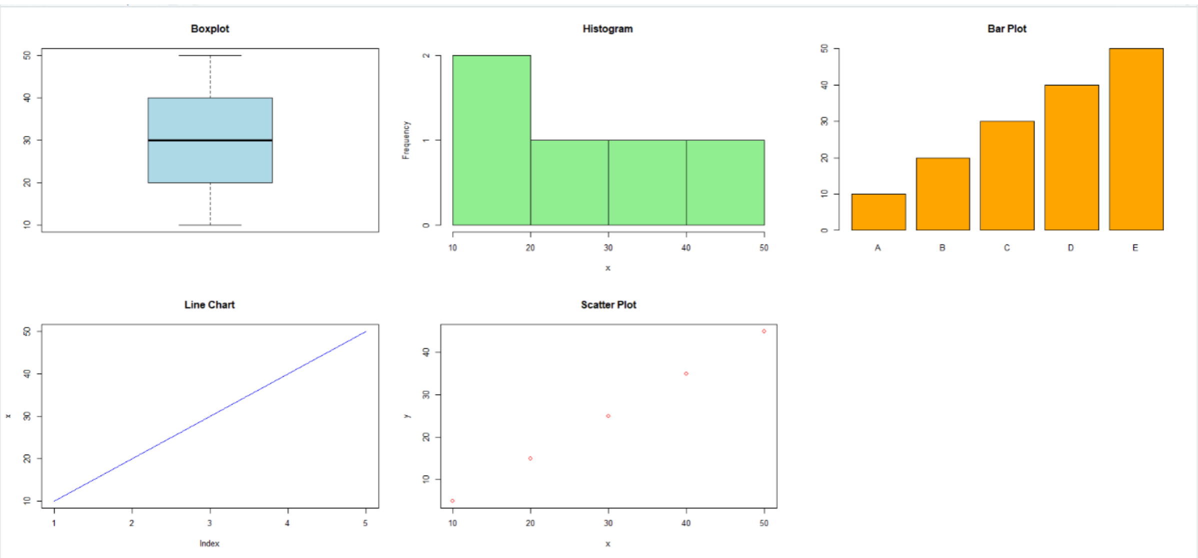
boxplot(x, main="Boxplot", col="lightblue")

hist(x, main="Histogram", col="lightgreen")

barplot(x, main="Bar Plot", col="orange", names.arg=c("A","B","C","D","E"))

plot(x, type="l", main="Line Chart", col="blue")

plot(x, y, main="Scatter Plot", col="red")



19. Build a regression model on advertising dataset (Sales ~ Spend) and predict Sales.

Spend <- c(10, 20, 30, 40, 50)

Sales <- c(15, 25, 35, 45, 55)

model <- lm(Sales ~ Spend)

predict(model, data.frame(Spend = 60))

```
 1
65
> print("Contents of the CSV file:")
```

20. Create multiple regression model using ChickWeight dataset with “Time” and “Diet” as predictors; predict weight and compute model error.

data("ChickWeight")

m <- lm(weight ~ Time + Diet, data = ChickWeight)

p <- predict(m)

mean((ChickWeight$weight - p)^2)

```
> mean((ChickWeight$weight - p)^2)
[1] 1284.319
> print("Contents of the CSV file:")
```

21. Randomly split iris dataset into train/test (80/20), build logistic regression (Species ~ Petal.Length + Petal.Width), predict, and evaluate with confusion matrix.

data("iris")

iris <- subset(iris, Species %in% c("setosa", "versicolor"))

m <- glm(Species ~ Petal.Length + Petal.Width, data=iris, family=binomial)

p <- ifelse(predict(m, type="response") > 0.5, "versicolor", "setosa")

table(p, iris$Species)

```
Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
> p <- ifelse(predict(m, type="response") > 0.5, "versicolor", "setosa")
> table(p, iris$Species)

p            setosa versicolor virginica
  setosa         50          0         0
  versicolor      0         50         0
> print("Contents of the CSV file:")
```