LAB

STATISTICS WITH R PROGRAMMING FOR VISUALIZATION

COURSE CODE: ITA0435

NAME:P.NIKITHA REG NO:192324226

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

CODE:

```
num_vec <- c(10, 20, 30)

char_vec <- c("A", "B", "C")

log_vec <- c(TRUE, FALSE, TRUE)

print(num_vec)

print(char_vec)

print(log_vec)

print(typeof(num_vec))

print(typeof(char_vec))

print(typeof(log_vec))
```

OUTPUT:

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

CODE:

```
mat1 <- matrix(1:20, nrow = 5, ncol = 4, byrow = TRUE)

rownames(mat1) <- paste("R", 1:5, sep="")

colnames(mat1) <- paste("C", 1:4, sep="")
```

```
mat1

mat2 <- matrix(1:9, nrow = 3, ncol = 3)

rownames(mat2) <- paste("R", 1:3, sep="")

colnames(mat2) <- paste("C", 1:3, sep="")

mat2

mat3 <- matrix(1:4, nrow = 2, ncol = 2, byrow = TRUE)

rownames(mat3) <- c("Row1", "Row2")

colnames(mat3) <- c("Col1", "Col2")

mat3
```

OUTPUT:

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

CODE:

```
rows <- 3

cols <- 3

tables <- 2

arr <- array(1:(rows * cols * tables),

             dim = c(rows, cols, tables))

print(arr)
```

OUTPUT:

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

CODE:

```
data_vec <- 1:12

arr <- array(data_vec,

          dim = c(2, 3, 2),

          dimnames = list(

            Row = c("R1", "R2"),

            Column = c("C1", "C2", "C3"),

            Table = c("T1", "T2")

          ))

print(arr)

print(arr["R1", "C2", "T1"])

print(arr["R2", "C3", "T2"])
```

OUTPUT:

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

CODE:

```
data(women)

height_group <- factor(

  ifelse(women$height < 65, "Short",

         ifelse(women$height <= 70, "Medium", "Tall"))

)

print(height_group)

levels(height_group)

set.seed(1)

letters_sample <- sample(LETTERS[1:5], 10, replace = TRUE)

letters_factor <- factor(letters_sample)

print(letters_factor)

levels(letters_factor)

table(letters_factor)
```

OUTPUT:

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

CODE:

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

mat <- matrix(1:9, nrow=3)

fun <- function(x) x^2

my_list <- list(Vector=vec, Matrix=mat, Function=fun)

print(my_list)

OUTPUT:

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.

CODE:

```
rand_vec <- sample(-50:50, 10, replace = TRUE)
print(rand_vec)
for(i in 1:100){
  if(i %% 3 == 0 && i %% 5 == 0){
    print("FizzBuzz")
  } else if(i %% 3 == 0){
    print("Fizz")
  } else if(i %% 5 == 0){
    print("Buzz")
  } else {
    print(i)
  }
}
```

OUTPUT:

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

CODE:

```
set.seed(1)
x <- rnorm(20, mean = 0, sd = 1)
```

print(x)

x_round <- round(x, 1)

freq <- table(x_round)

print(freq)

OUTPUT:

9. Create empty plots with specified axis limits.

CODE:

plot(1, type = "n",

xlim = c(0, 10),

ylim = c(0, 20),

xlab = "X-Axis",

ylab = "Y-Axis",

main = "Empty Plot")

OUTPUT:

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

CODE:

```
exam_data <- data.frame(
  name = c("niki", "navya", "Chandu", "Divya"),
  score = c(85, 72, 90, 65),
  attempts = c(1, 2, 1, 3),
  qualify = c(TRUE, TRUE, TRUE, FALSE)
)
print(exam_data)
print(exam_data$score)
print(exam_data[1:2, ])
new_row <- data.frame(
  name = "Esha",
  score = 88,
  attempts = 1,
  qualify = TRUE
)
exam_data <- rbind(exam_data, new_row)
exam_data$grade <- c("B", "C", "A", "D", "B")
print(exam_data)
```

sorted_data <- exam_data[order(-exam_data$score),]

print(sorted_data)

write.csv(exam_data, "exam_data.csv", row.names = FALSE)

OUTPUT:

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

CODE:

data <- read.csv("exam_data.csv")

print(data)

OUTPUT:

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

CODE:

```
data(airquality)
monthly_avg <- aggregate(
  cbind(Ozone, Solar.R, Wind, Temp) ~ Month,
  data = airquality,
  FUN = mean,
  na.rm = TRUE
)
print(monthly_avg)
```

OUTPUT:

13. Combine multiple arrays row-wise.

CODE:

```
a1 <- matrix(1:6, nrow=2)
a2 <- matrix(7:12, nrow=2)
combined <- rbind(a1, a2)
print(combined)
```

OUTPUT:

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

CODE:

```
data(ChickWeight)

head(ChickWeight)

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

library(reshape2)

melted <- melt(ChickWeight, id=c("Chick","Diet"))

dcast(melted, Diet ~ variable, mean)
```

OUTPUT:

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

CODE:

```
data(iris)
dim(iris)
summary(iris)
sd(iris$Sepal.Length)
quantile(iris$Sepal.Length)
aggregate(. ~ Species, iris, mean)
table(cut(iris$Sepal.Length, breaks=3), iris$Species)
```

OUTPUT:

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.

CODE:

```
data(USArrests)
summary(USArrests)
rownames(USArrests)[which.max(USArrests$Rape)]
max(USArrests$Murder); min(USArrests$Murder)
cor(USArrests)
USArrests[USArrests$Assault > median(USArrests$Assault),]
```

USArrests[USArrests$Murder < quantile(USArrests$Murder,0.25),]

barplot(USArrests$Rape, names.arg=rownames(USArrests), las=2)

hist(USArrests$Murder)

plot(density(USArrests$Assault))

plot(USArrests$Murder, USArrests$Assault)

OUTPUT:

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

CODE:

```
data(Titanic)

titanic_df <- as.data.frame(Titanic)

barplot(table(titanic_df$Survived, titanic_df$Class),

        beside=TRUE,

        col=c("red","green"),

        legend=TRUE)
```

```
barplot(table(titanic_df$Survived, titanic_df$Sex),
        beside=TRUE,
        col=c("blue","pink"),
        legend=TRUE)
```

OUTPUT:

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

CODE:

```
boxplot(iris$Sepal.Length)

hist(iris$Petal.Length)
```

barplot(table(iris$Species))

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

plot(iris$Sepal.Length, iris$Petal.Length)

OUTPUT:

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled33*
r program activity 1.R
r program activity 2.R
r program lab.R*
Source on Save
Run
Source
col=c("blue","pink"),
legend=TRUE)
boxplot(iris$Sepal.Length)
hist(iris$Petal.Length)
barplot(table(iris$Species))
plot(1:10, type="l")
plot(iris$Sepal.Length, iris$Petal.Length)
242:1 (Top Level)
R Script
Console
Terminal
Jobs
R 4.4.3 · ~/
Minnesota 2.7 72 66 14.9
New Hampshire 2.1 57 56 9.5
North Dakota 0.8 45 44 7.3
Rhode Island 3.4 174 87 8.3
South Dakota 3.8 86 45 12.8
Utah 3.2 120 80 22.9
Vermont 2.2 48 32 11.2
Washington 4.0 145 73 26.2
Wisconsin 2.6 53 66 10.8
> barplot(USArrests$Rape, names.arg=rownames(USArrests), las=2)
> hist(USArrests$Murder)
> plot(density(USArrests$Assault))
> plot(USArrests$Murder, USArrests$Assault)
> data(Titanic)
> titanic_df <- as.data.frame(Titanic)
> barplot(table(titanic_df$Survived, titanic_df$Class),
+ beside=TRUE,
+ col=c("red","green"),
+ legend=TRUE)
> barplot(table(titanic_df$Survived, titanic_df$Sex),
+ beside=TRUE,
+ col=c("blue","pink"),
+ legend=TRUE)
> barplot(table(titanic_df$Survived, titanic_df$Sex),
+ beside=TRUE,
+ col=c("blue","pink"),
+ legend=TRUE)
> data(Titanic)
> titanic_df <- as.data.frame(Titanic)
> barplot(table(titanic_df$Survived, titanic_df$Class),
+ beside=TRUE,
+ col=c("red","green"),
+ legend=TRUE)
> boxplot(iris$Sepal.Length)
> hist(iris$Petal.Length)
> barplot(table(iris$Species))
R Graphics: Device 2 (ACTIVE)
File History Resize
0
10
20
30
40
50
setosa
versicolor
virginica

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled33*
r program activity 1.R
r program activity 2.R
r program lab.R*
Source on Save
Run
Source
col=c("blue","pink"),
legend=TRUE)
boxplot(iris$Sepal.Length)
hist(iris$Petal.Length)
barplot(table(iris$Species))
plot(1:10, type="l")
plot(iris$Sepal.Length, iris$Petal.Length)
243:1 (Top Level)
R Script
Console
Terminal
Jobs
R 4.4.3 · ~/
New Hampshire 2.1 57 56 9.5
North Dakota 0.8 45 44 7.3
Rhode Island 3.4 174 87 8.3
South Dakota 3.8 86 45 12.8
Utah 3.2 120 80 22.9
Vermont 2.2 48 32 11.2
Washington 4.0 145 73 26.2
Wisconsin 2.6 53 66 10.8
> barplot(USArrests$Rape, names.arg=rownames(USArrests), las=2)
> hist(USArrests$Murder)
> plot(density(USArrests$Assault))
> plot(USArrests$Murder, USArrests$Assault)
> data(Titanic)
> titanic_df <- as.data.frame(Titanic)
> barplot(table(titanic_df$Survived, titanic_df$Class),
+ beside=TRUE,
+ col=c("red","green"),
+ legend=TRUE)
> barplot(table(titanic_df$Survived, titanic_df$Sex),
+ beside=TRUE,
+ col=c("blue","pink"),
+ legend=TRUE)
> barplot(table(titanic_df$Survived, titanic_df$Sex),
+ beside=TRUE,
+ col=c("blue","pink"),
+ legend=TRUE)
> data(Titanic)
> titanic_df <- as.data.frame(Titanic)
> barplot(table(titanic_df$Survived, titanic_df$Class),
+ beside=TRUE,
+ col=c("red","green"),
+ legend=TRUE)
> boxplot(iris$Sepal.Length)
> hist(iris$Petal.Length)
> barplot(table(iris$Species))
> plot(1:10, type="l")
R Graphics: Device 2 (ACTIVE)
File History Resize
1:10
2
4
6
8
10
Index

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

CODE:

```
advertising <- data.frame(
  Spend = c(100,150,200,250,300),
  Sales = c(10,15,22,28,35)
)
model <- lm(Sales ~ Spend, data=advertising)
summary(model)
predict(model, data.frame(Spend=200))
```

OUTPUT:

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

CODE:

data(ChickWeight)

model2 <- lm(weight ~ Time + Diet, data=ChickWeight)

pred <- predict(model2, ChickWeight)

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

OUTPUT:

21. Randomly split iris dataset into train/test (80/20), build logistic regression (Species ~ Petal.Length +

Petal.Width), predict, and evaluate with confusion matrix

CODE:

```
set.seed(1)

data(iris)

iris2 <- subset(iris, Species != "setosa")

iris2$Species <- factor(iris2$Species)

train_index <- sample(1:nrow(iris2), 0.8*nrow(iris2))

train <- iris2[train_index,]

test <- iris2[-train_index,]

model <- glm(Species ~ Petal.Length + Petal.Width,

             data=train, family=binomial)

prob <- predict(model, test, type="response")

pred <- ifelse(prob > 0.5,

               levels(train$Species)[2],

               levels(train$Species)[1])

table(Predicted=pred, Actual=test$Species)
```

OUTPUT: