UCSC

SCS2211 - LABORATORY II
R Lab Practical Sheet - 10

Instructions

- **Do the Activities and save in a .ipynb file**
- **File name should be <Index number>.rmd (Eg: 2000000.ipynb) and upload to the given link.**
- **Any form of plagiarism or collusion is not allowed**

Control Structures in RStudio

Control structures in R allow us to control the flow of execution of a program based on conditions or iterations. They are essential for decision-making, repetitive tasks, and executing specific blocks of code conditionally. Below are the common control structures in R, along with examples and related codes.

1. **if Statement**

   The if statement is used to execute a block of code if a specified condition is true.

   Example:

   ```
   x <- 10
   if (x > 5) {
       print("x is greater than 5")
   }
   ```

2. **if-else Statement**

   The if-else statement allows for an alternate block of code to execute if the condition is false.

   Example:

   ```
   y <- 3
   if (y %% 2 == 0) {
       print("y is even")
   } else {
       print("y is odd")
   }
   ```

3. **ifelse Function**

   The ifelse function is a vectorized version of the if-else statement.

   Example:

   ```
   values <- c(5, 8, 12, 3, 7)
   result <- ifelse(values > 6, "Greater", "Smaller")
   print(result)
   ```

4. **for Loop**

   The for loop is used for iterating over a sequence or vector.

   Example:

```
for (i in 1:5) {
    print(paste("Iteration:", i))
}
```

5. **while Loop**

   The while loop is used to execute a block of code as long as the condition is true.

   Example:

```
count <- 1
while (count <= 5) {
    print(paste("Count is:", count))
    count <- count + 1
}
```

6. **repeat Loop**

   The repeat loop executes indefinitely unless explicitly stopped using a break statement.

   Example:

```
num <- 1
repeat {
    print(num)
    if (num == 5) {
        break
    }
    num <- num + 1
}
```

7. **break Statement**

   The break statement is used to exit a loop prematurely.

   Example:

```
for (i in 1:10) {
    if (i == 6) {
        break
    }
    print(i)
}
```

8. **next Statement**

   The next statement is used to skip the current iteration and proceed to the next one.

   Example:

```
for (i in 1:10) {
    if (i %% 2 == 0) {
        next
    }
    print(i)
}
```

9. **Nested Loops**

   Loops can be nested to handle more complex iterations.

Example:

```
for (i in 1:3) {
    for (j in 1:2) {
        print(paste("i =", i, ", j =", j))
    }
}
```

10. **Custom Function with Control Structures**

    Control structures can also be used within functions to perform specific operations.

    Example:

```
evaluate_number <- function(num) {
    if (num > 0) {
        return("Positive")
    } else if (num < 0) {
        return("Negative")
    } else {
        return("Zero")
    }
}
print(evaluate_number(10))
print(evaluate_number(-5))
print(evaluate_number(0))
```

Practice Task:

Write a script to calculate the factorial of a number using a for loop.

Solution:

```
factorial <- 1
num <- 5
for (i in 1:num) {
    factorial <- factorial * i
}
print(paste("Factorial of", num, "is", factorial))
```

Dataset Analysis with Control Structures

Dataset: https://www.kaggle.com/uciml/iris

- **How to Load the dataset:**

```
data <- read.csv("iris.csv")
```

- **How to Find the number of records:**

```
count <- 0
for (i in 1:nrow(data)) {
    count <- count + 1
}
print(paste("Number of records:", count))
```

- **How to Print the summary of a column:**

```
min_val <- min(data$Sepal.Width)
max_val <- max(data$Sepal.Width)
```

```
mean_val <- mean(data$Sepal.Width)
summary <- paste("Min:", min_val, "Max:", max_val, "Mean:", mean_val)
print(summary)
```

- **Find the number of rows with Sepal.Width greater than the mean:**

```
mean_val <- mean(data$Sepal.Width)
count <- 0
for (i in 1:nrow(data)) {
    if (data$Sepal.Width[i] > mean_val) {
        count <- count + 1
    }
}
print(paste("Rows with Sepal.Width > mean:", count)):
```

- **Find unique values:**

```
unique_species <- unique(data$Species)
print(unique_species)
```

- **Dataset Analysis with Control Structures**

  Print Sepal.Length of the first 10 rows where Sepal.Width > 3.5:

```
Count <- 0
for (i in 1:nrow(data)) {
    if (data$Sepal.Width[i] > 3.5 && count < 10) {
        print(data$Sepal.Length[i])
        count <- count + 1
    }
}
```

  Function to check petal length status:

```
check_petal_length <- function(row) {
    if (row$Petal.Length < 3.0) {
        return("Low")
    } else {
        return("High")
    }
}
for (i in 1:nrow(data)) {
    row <- data[i, ]
    print(check_petal_length(row))
}
```

Pie Chart and Bar Chart Representation in RStudio

R provides excellent support for data visualization, including the creation of pie charts and bar charts. Below is a guide to help you create and customize these charts with examples and related code.

Pie Chart Representation

A pie chart is used to represent data as slices of a circle, where each slice represents a proportion of the whole.

Example 1: Basic Pie Chart

```
# Data
labels <- c("Category A", "Category B", "Category C", "Category D")
values <- c(25, 15, 30, 30)
# Pie Chart
pie(values, labels = labels, main = "Basic Pie Chart", col = rainbow(length(labels)))
```

Example 2: Pie Chart with Percentages

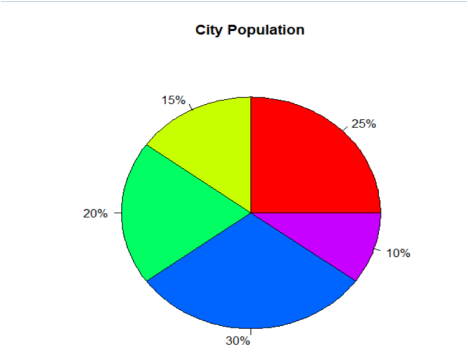
```
# Calculate percentages
percentages <- round(values / sum(values) * 100)
labels_with_percent <- paste(labels, percentages, "%", sep = " ")
# Pie Chart
pie(values, labels = labels_with_percent, main = "Pie Chart with Percentages", col =
rainbow(length(labels)))
```

Example 3: Customized Pie Chart

```
# Data
values <- c(40, 20, 30, 10)
labels <- c("A", "B", "C", "D")
# Custom Pie Chart
pie(values, labels = labels, main = "Customized Pie Chart", col = c("red", "blue",
"green", "yellow"))
```

Example 4: City Population Pie Chart

```
# Data
cities <- c("City A", "City B", "City C", "City D", "City E")
population <- c(50000, 30000, 40000, 60000, 20000)
# Pie Chart using Rainbow Palette
pie(population, labels = cities, col = rainbow(length(cities)), main = "City
Population")
# Pie Chart with Percentages
percentages <- round(population / sum(population) * 100)
labels <- paste(percentages, "%", sep = "")
pie(population, labels = labels, col = rainbow(length(cities)), main = "City
Population")
```



Bar Chart Representation

A bar chart is used to represent data with rectangular bars whose lengths are proportional to the values they represent.

Example 1: Basic Bar Chart

```
# Data
years <- c("2019", "2020", "2021", "2022")
values <- c(500, 700, 600, 800)
# Bar Chart
barplot(values, names.arg = years, main = "Basic Bar Chart", col = "lightblue")
```

Example 2: Bar Chart with Customization

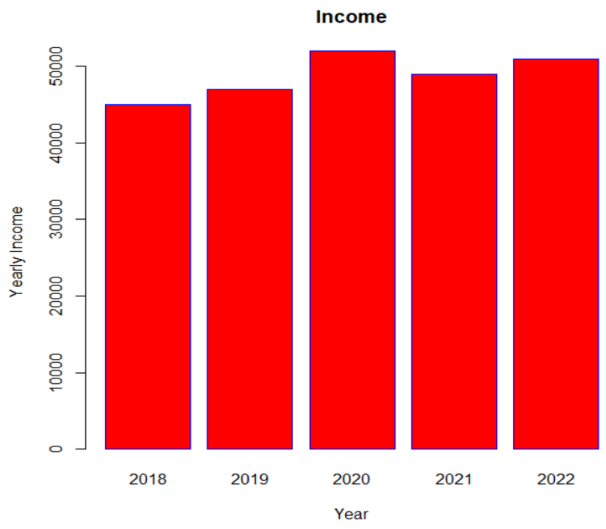
```
# Customized Bar Chart
barplot(values, names.arg = years, col = "orange", border = "brown",
        xlab = "Year", ylab = "Values", main = "Customized Bar Chart")
```

Example 3: Horizontal Bar Chart

```
# Horizontal Bar Chart
barplot(values, names.arg = years, horiz = TRUE, col = "purple", border = "black",
        xlab = "Values", ylab = "Year", main = "Horizontal Bar Chart")
```

Example 4: Yearly Income Bar Chart

```
# Data
years <- c("2018", "2019", "2020", "2021", "2022")
income <- c(45000, 47000, 52000, 49000, 51000)
# Bar Chart
barplot(income, names.arg = years, col = "red", border = "blue",
        xlab = "Year", ylab = "Yearly Income", main = "Income")
```



Activity 01

DataSet : https://www.kaggle.com/fedesoriano/heart-failure-prediction

While following tasks can be easily done using specific commands in R you are expected to achieve these results using control structures

1. Load the dataset
2. Find the number of records
3. Print the summary of RestingBP
4. Find the number of patients with “higher than mean” RestingBP.
5. Find unique chest pain types.
6. Print the ages of the first 10 Females whose resting BP is greater than 140.
7. Write a function which takes a row of data as input and outputs the attribute
8. information in a structured manner. If cholesterol level is less than 200 then print low else high.

Activity 02

1. Consider the following Table

City	Population
Colombo	580000
Kandy	450000
Galle	330000
Anuradhapura	380000
Trincomalee	280000
Jaffna	310000

2. Represent the above table in a pie chart
   - Using the Rainbow color palette
   - Title of the pie chart - “City Population”

3. Represent the above pie chart, where it shows the percentage except the city as Labels.

Activity 03

1. Consider the following table

Year	Yearly income
2015	1 320 000
2016	1 500 000
2017	2 002 000
2018	1 980 000
2019	1 520 000

2. Create a bar chart for the above table.
   - Name the y axis as “Yearly Income”, x axis as “Year”
   - Title - Income
   - Bar color - red, outline - blue