R Language: Subsetting, Input, and Output

1. Subsetting in R

Subsetting is used to access specific parts of data structures like vectors, matrices, lists, and data frames.

- **Subsetting Vectors**: Use the index to access specific elements.

```
# Creating a vector
vector <- c(10, 20, 30, 40, 50)

# Extracting a specific element (2nd element)
second_element <- vector[2]  # 20

# Extracting multiple elements (1st and 3rd elements)
subset_vector <- vector[c(1, 3)]  # 10, 30
```

- **Subsetting Matrices**: Use row and column indices.

```
# Creating a matrix
matrix_data <- matrix(1:9, nrow = 3, ncol = 3)

# Extracting element at 2nd row and 3rd column
matrix_element <- matrix_data[2, 3]  # 6

# Extracting 1st row
first_row <- matrix_data[1, ]  # 1, 2, 3

# Extracting 2nd column
second_column <- matrix_data[, 2]  # 2, 5, 8
```

- **Subsetting Lists**: Use double square brackets to access elements.

```
# Creating a list
my_list <- list("Apple", 42, TRUE)

# Extracting the 2nd element (number)
second_element_list <- my_list[[2]]  # 42
```

- **Subsetting Data Frames**: Use column names or row/column indices.

```
# Creating a data frame
df <- data.frame(Name = c("Alice", "Bob", "Charlie"),
                 Age = c(25, 30, 35),
                 Gender = c("Female", "Male", "Male"))

# Extracting the "Name" column
name_column <- df$Name  # "Alice", "Bob", "Charlie"

# Extracting the 1st row
first_row_df <- df[1, ]  # "Alice", 25, "Female"
```

2. Input in R

In R, we can accept input using functions like `readline()` for user input or `scan()` for reading data from the console.

- **Using readline()**: To accept text input.

```
# Reading a single string input
name_input <- readline(prompt = "Enter your name: ")
# Example input: "John"

# Output the result
print(paste("Hello,", name_input))
```

- **Using scan()**: To read numeric or other types of data.

```
# Reading multiple numbers
numbers_input <- scan()

# Example input: 1 2 3 4 5

# Output the sum of numbers
sum_input <- sum(numbers_input)
print(paste("The sum is:", sum_input))
```

3. Output in R

In R, the output can be displayed using the `print()` function or simply by typing the expression.

- **Using print()**: Explicitly print a value.

```
# Printing a string
print("Hello, R!")

# Printing a numeric value
num <- 10
print(num)

# Printing a data frame
df <- data.frame(Name = c("Alice", "Bob", "Charlie"),
                 Age = c(25, 30, 35),
                 Gender = c("Female", "Male", "Male"))
print(df)
```

- **Implicit Output**: R also outputs the result of an expression without needing `print()`.

```
# Assign a value and implicitly output it
result <- 10 + 5  # R will output 15

# Working with complex output (matrix, data frame)
matrix_result <- matrix(1:6, nrow = 2)
matrix_result  # Implicit output
```