Unit 3: R packages and functions

Building R Packages, Installing and loading packages, Running and Manipulating Packages, Setting up your working directory, Downloading and importing data, working with objects, Viewing Objects within Objects, Constructing Data Objects, Functions in R, Creating functions, calling functions, Writing R scripts.

1. Building R Packages

R packages are collections of R functions, data, and documentation bundled together. You can build a package to reuse or share your code.

```
install.packages("devtools")      # Installs devtools to help build packages
library(devtools)                 # Load devtools

create("MyFirstPackage")          # Creates a new package structure
```

Output:

✓ Creating 'MyFirstPackage/'
✓ Writing 'MyFirstPackage/DESCRIPTION'
✓ Writing 'MyFirstPackage/NAMESPACE'
✓ Setting active project to '.../MyFirstPackage'

2. Installing and Loading Packages

Description:
Packages extend R's capabilities. You need to install them once and load them every time you want to use them.

```
install.packages("ggplot2")  # Install ggplot2 package
library(ggplot2)             # Load the package
```

Output:

Installing package into 'C:/...'
package 'ggplot2' successfully unpacked

3. Running and Manipulating Packages

Description:
Once a package is loaded, you can call its functions. For example, ggplot2 provides visualization functions.

```
qplot(mpg, wt, data = mtcars)  # Create a quick scatter plot
```

Output:
A scatter plot of miles-per-gallon (mpg) vs weight (wt) from the mtcars dataset.

4. Setting Up Your Working Directory

Description:
The working directory is where R reads and saves files. Setting it properly avoids file-not-found errors.

```
getwd()                          # View current directory
setwd("C:/Users/YourName/RWork")  # Set a new working directory
```

Output:

```
[1] "C:/Users/YourName/RWork"
```

5. Downloading and Importing Data

Description:
You can load external datasets into R using functions like read.csv().

```
data <- read.csv("https://people.sc.fsu.edu/~jburkardt/data/csv/airtravel.csv")
head(data)
```

Output:

```
  "Month"  "1958"  "1959"  "1960"
1 JAN      340     360     417
2 FEB      318     342     391
3 MAR      362     406     419
```

6. Working with Objects

Description:
Everything in R is an object—vectors, lists, data frames, etc. You can assign and manipulate them easily.

```
x <- 10
names <- c("A", "B")
scores <- c(85, 90)
```

```
students <- data.frame(names, scores)

print(students)
```

Output:

```
  names scores
1     A     85
2     B     90
```

7. Viewing Objects within Objects

Description:
To understand complex objects, use str(), names(), or summary().

```
str(students)        # Structure
names(students)      # Column names
summary(students)    # Statistical summary
```

Output:

```
'data.frame':           2 obs. of  2 variables:
 $ names : chr  "A" "B"
 $ scores: num  85 90
[1] "names"  "scores"
    names   scores
 A:1     Min.   :85.0
 B:1     Max.   :90.0
```

8. Constructing Data Objects

Description:
R supports different types: vectors, matrices, lists, data frames.

```
vec <- c(1, 2, 3)
mat <- matrix(1:6, nrow=2)
lst <- list(Name="Ravi", Age=25)
df <- data.frame(ID=1:2, Marks=c(80, 90))

print(mat)
```

Output:

```
     [,1] [,2] [,3]
```

```
[1,]  1  3  5
[2,]  2  4  6
```

9. Functions in R

Description:
Functions are built-in tools that perform tasks. Example: sum(), mean(), etc.

```
sum(c(1,2,3))     # Adds numbers
mean(c(10,20))    # Calculates mean
sqrt(16)          # Square root
```

Output:

```
[1] 6
[1] 15
[1] 4
```

10. Creating Functions

Description:
You can write your own functions using function().

```
add_numbers <- function(a, b) {
  return(a + b)
}
```

Output:
(No output when defining.)

---

11. Calling Functions

Description:
Once defined, functions can be called using their name and passing arguments.

```
result <- add_numbers(5, 3)
print(result)
```

Output:

```
[1] 8
```

---

12. Writing R Scripts

Description:
You can write multiple lines of R code in .R files and run them using source().

■ *myscript.R contents:*

```
a <- 4
b <- 5
cat("Sum is:", a + b)
source("myscript.R")
```

Output:

```
Sum is: 9
```

Would you like this as a **printable PDF or PPT** version with visuals? I can generate it for you.