Challenge-3

Soh Li Ying

2023-08-28

I. Questions

Question 1: Emoji Expressions Imagine you're analyzing social media posts for sentiment analysis. If you were to create a variable named "postSentiment" to store the sentiment of a post using emojis (for positive, for neutral, for negative), what data type would you assign to this variable? Why? (*narrative type question, no code required*)

Solution: I would assign this variable to be of character data type since they have natural ordering and are also of non-numerical data type.

Question 2: Hashtag Havoc In a study on trending hashtags, you want to store the list of hashtags associated with a post. What data type would you choose for the variable "postHashtags"? How might this data type help you analyze and categorize the hashtags later? (*narrative type question, no code required*)

Solution:
The most suitable data type is Character since Character vectors are text strings, and hastags are also text strings. Hashtags that are of the same can be classified into a single category. In addition, we can also keep count of the number of appearances or repetition of these hashtags, thereby allowing us to analyse the popularity of each hashtags.

Question 3: Time Traveler's Log You're examining the timing of user interactions on a website. Would you use a numeric or non-numeric data type to represent the timestamp of each interaction? Explain your choice (*narrative type question, no code required*)

Solution: I will use a non-numeric data type to represent the timestamp of each interaction since these timestamps will usually contain the date and time of interations. A non-numeric data type will be able to represent each timestamp more precisely and with more details.

Question 4: Event Elegance You're managing an event database that includes the date and time of each session. What data type(s) would you use to represent the session date and time? (*narrative type question, no code required*)

Solution:
I will use non-numeric data types such as characters to represent the session date and time since characters can be of strings containing both alphabets or numerals.

Question 5: Nominee Nominations You're analyzing nominations for an online award. Each participant can nominate multiple candidates. What data type would be suitable for storing the list of nominated candidates for each participant? (*narrative type question, no code required*)

Solution: The suitable data type would be character data type since the list of nominated candidates would contain names that will be of characters data type.

Question 6: Communication Channels In a survey about preferred communication channels, respondents choose from options like "email," "phone," or "social media." What data type would you assign to the variable "preferredChannel"? (*narrative type question, no code required*)

Solution: Since the given options of "email", "phone", or "social media" do not have a natural ordering, and neither are they numerical variables, I will assign "preferredChannel" to be as Character date type.

Question 7: Colorful Commentary In a design feedback survey, participants are asked to describe their feelings about a website using color names (e.g., "warm red," "cool blue"). What data type would you choose for the variable "feedbackColor"? (*narrative type question, no code required*)

Solution: Since the variable will store color names that are of text strings, I will chose Character data type as a suitable type for the variable.

Question 8: Variable Exploration Imagine you're conducting a study on social media usage. Identify three variables related to this study, and specify their data types in R. Classify each variable as either numeric or non-numeric.

Solution: The first variable related to this study is "Time Spent Online", that is of numeric variable and a double data type. The second variable is "Preferred Social Media Platform". It is a non-numeric variable and of character data type. The third variable is "Post Engagement Rate". It is a numeric variable, and of integer data type.

Question 9: Vector Variety Create a numeric vector named "ages" containing the ages of five people: 25, 30, 22, 28, and 33. Print the vector.

Solution:

```r
ages <- c(25, 30, 22, 28, 33)
print(ages)
```

```
## [1] 25 30 22 28 33
```

Question 10: List Logic Construct a list named "student_info" that contains the following elements:

- A character vector of student names: "Alice," "Bob," "Catherine"
- A numeric vector of their respective scores: 85, 92, 78
- A logical vector indicating if they passed the exam: TRUE, TRUE, FALSE

Print the list.

Solution:

```r
# Construct list
student_info <- list( names = c("Alice", "Bob", "Catherine"),
                      scores = c(85, 92, 78),
                      passed_exam = c(TRUE, TRUE, FALSE))

print(student_info)
```

```
## $names
## [1] "Alice"     "Bob"       "Catherine"
##
## $scores
## [1] 85 92 78
##
## $passed_exam
## [1]  TRUE  TRUE FALSE
```

Question 11: Type Tracking You have a vector "data" containing the values 10, 15.5, "20", and TRUE. Determine the data types of each element using the typeof() function.

Solution:

```
# Create vector
data <- c(10, 15.5, "20", TRUE)

# Determine data types of each element
typeof(data[1])
```

```
## [1] "character"
```

```
typeof(data[2])
```

```
## [1] "character"
```

```
typeof(data[3])
```

```
## [1] "character"
```

```
typeof(data[4])
```

```
## [1] "character"
```

Question 12: Coercion Chronicles You have a numeric vector "prices" with values 20.5, 15, and "25". Use explicit coercion to convert the last element to a numeric data type. Print the updated vector.

Solution:

```
# Constract vector
prices <- c(20.5, 15, "25")

# Use explicit coercion to convert last element to numeric data type
prices <- as.numeric(prices)
print(prices)
```

```
## [1] 20.5 15.0 25.0
```

Question 13: Implicit Intuition Combine the numeric vector c(5, 10, 15) with the character vector c("apple", "banana", "cherry"). What happens to the data types of the combined vector? Explain the concept of implicit coercion.

Solution:

```r
# Construct vectors
numeric_vector <- c(5, 10, 15)
character_vector <- c("apple", "banana", "cherry")

# Combine vectors
combined_vector <- c(numeric_vector, character_vector)

# Identify data types of combined vector
typeof(combined_vector)
```

```
## [1] "character"
```

R will convert the elements of the vector to the type based on its content. So from the example above, since elements of character types are added to the vector, the data types of the combined vector will all be converted to be of character type as well.

Question 14: Coercion Challenges You have a vector "numbers" with values 7, 12.5, and "15.7". Calculate the sum of these numbers. Will R automatically handle the data type conversion? If not, how would you handle it?

Solution:

Since vector contains a mixture of numeric and character values, R will not be able to handle the data type conversion. The character "15.7" will be coerced to NA.

```r
# Construct vector
numbers <- c(7, 12.5, "15.7")

# Convert character value to numeric value
numbers <- as.numeric(numbers)

# Sum and print sum of numbers
sum_numbers <- sum(numbers)
print(sum_numbers)
```

```
## [1] 35.2
```

Question 15: Coercion Consequences Suppose you want to calculate the average of a vector "grades" with values 85, 90.5, and "75.2". If you directly calculate the mean using the mean() function, what result do you expect? How might you ensure accurate calculation?

Solution:

If you directly calculate the mean using the mean() function, an error will occur due to data type conversion issues. R will return result to NA.

```r
grades <- c(85, 90.5, "75.2")
mean_grade <- mean(grades)
```

```
## Warning in mean.default(grades): argument is not numeric or logical: returning
## NA
```

```r
print(mean_grade)
```

```
## [1] NA
```

To ensure accurate calculation, the character value “75.2” has to be explicitly converted to a numeric value first

```r
#Convert character values to numeric
grades <- as.numeric(grades)

#Calculate and print accurate mean results
mean_grade <- mean(grades)
print(mean_grade)
```

```
## [1] 83.56667
```

Question 16: Data Diversity in Lists Create a list named “mixed_data” with the following components:

- A numeric vector: 10, 20, 30
- A character vector: “red”, “green”, “blue”
- A logical vector: TRUE, FALSE, TRUE

Calculate the mean of the numeric vector within the list.

Solution:

```r
# Creating list
mixed_data <- list(numeric_vector = c(10, 20, 30),
                   character_vector = c("red", "green", "blue"),
                   logical_vector = c(TRUE, FALSE, TRUE))

# Calculate mean of numeric vector within list
numeric_mean <- mean(mixed_data$numeric_vector)
print(numeric_mean)
```

```
## [1] 20
```

Question 17: List Logic Follow-up Using the “student_info” list from Question 10, extract and print the score of the student named “Bob.”

Solution:

```r
# Extract and print score of Bob
student_info
```

```
## $names
## [1] "Alice"     "Bob"       "Catherine"
##
## $scores
## [1] 85 92 78
##
## $passed_exam
## [1]  TRUE  TRUE FALSE
```

```r
bob_score <- student_info$scores[student_info$names == "Bob"]
print(bob_score)
```

```
## [1] 92
```

Question 18: Dynamic Access Create a numeric vector values with random values. Write R code to dynamically access and print the last element of the vector, regardless of its length.

Solution:

```r
# Generate a numeric vector with random values
values <- c(23, 19, 212, 3592, 14, 1712)

# Access and print last element
last_element <- values[length(values)]
print(last_element)
```

```
## [1] 1712
```

Question 19: Multiple Matches You have a character vector words <- c("apple", "banana", "cherry", "apple"). Write R code to find and print the indices of all occurrences of the word "apple."

Solution:

```r
# Generate character vector
words <- c("apple", "banana", "cherry", "apple")

#Print indices of all occurrences of word "apple"
which(words == "apple")
```

```
## [1] 1 4
```

Question 20: Conditional Capture Assume you have a vector ages containing the ages of individuals. Write R code to extract and print the ages of individuals who are older than 30.

Solution:

```r
# Generate vector containing age of individuals
ages <- c(13, 53, 72, 65, 27)

# Print & extract the ages of individuals > 30
older <- ages[ages > 30]
print(older)
```

```
## [1] 53 72 65
```

Question 21: Extract Every Nth Given a numeric vector sequence <- 1:20, write R code to extract and print every third element of the vector.

Solution:

```r
# Generate numeric vector sequence from 1 to 20
data <- seq(from = 1, to = 20, by = 1)

#Extract and print every third element of the vector
every_third_element <- data[seq(from = 3, to = 20, by = 3)]
print(every_third_element)
```

```
## [1]  3  6  9 12 15 18
```

Question 22: Range Retrieval Create a numeric vector numbers with values from 1 to 10. Write R code to extract and print the values between the fourth and eighth elements.

Solution:

```r
# Generate numeric vector numbers from 1 to 10
data <- seq(from = 1, to = 10, by = 1)

# Print values between forth and eighth elements
data[4:8]
```

```
## [1] 4 5 6 7 8
```

Question 23: Missing Matters Suppose you have a numeric vector data <- c(10, NA, 15, 20). Write R code to check if the second element of the vector is missing (NA).

Solution:

```r
# Generate numeric vector data
data <- c(10, NA, 15, 20)

# Check if second element of vector is missing (NA)
missing <- is.na(data[2])
print(missing)
```

```
## [1] TRUE
```

Question 24: Temperature Extremes Assume you have a numeric vector temperatures with daily temperatures. Create a logical vector hot_days that flags days with temperatures above 90 degrees Fahrenheit. Print the total number of hot days.

Solution:

```r
# Generate numeric vector with daily temperatures
temperature <- c(90, 100, 59, 95, 91)

# Extract values of temperatures > 90 degrees Fahrenheit
hot_days <- temperature > 90

# Sum and print total number of days
total_number_of_hot_days <- sum(hot_days)
print(total_number_of_hot_days)
```

```
## [1] 3
```

Question 25: String Selection Given a character vector fruits containing fruit names, create a logical vector long_names that identifies fruits with names longer than 6 characters. Print the long fruit names.

Solution:

```r
# Generate a character vector containing fruit names
fruit_names <- c("apple", "cherry", "banana", "pineapple", "watermelon", "mangosteen")

# Extract and print name of fruits with names longer than 6 characters
long_names <- nchar(fruit_names) > 6
print(fruit_names[long_names])
```

```
## [1] "pineapple"  "watermelon" "mangosteen"
```

Question 26: Data Divisibility Given a numeric vector numbers, create a logical vector divisible_by_5 to indicate numbers that are divisible by 5. Print the numbers that satisfy this condition.

Solution:

```r
# Generate a numeric vector
numeric_vector <- c(5, 10, 17, 34, 35)

# Extract and print numbers that are divisible by 5
divisible_by_5 <- numeric_vector %% 5 == 0
print(numeric_vector[divisible_by_5])
```

```
## [1]  5 10 35
```

Question 27: Bigger or Smaller? You have two numeric vectors vector1 and vector2. Create a logical vector comparison to indicate whether each element in vector1 is greater than the corresponding element in vector2. Print the comparison results.

Solution:

```r
# Generate vector1 and vector2
vector1 <- c(10, 6, 7)
vector2 <- c(20, 3, 17)

# Compare and print results of whether each element in vector1 is greater than corresponding element in
compare_vectors <- vector1 > vector2
print(compare_vectors)
```

```
## [1] FALSE  TRUE FALSE
```