Challenge-3

Marzuki Noonanas

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: The variable "postSentiment" indicates the response of the social media post - an Ordinal Categorical variable with a natural order. In this case, I can represent the values of 😊 to be 1, 😐 to be 0 and 😞 to be -1. This allows flexibility where data could be analysed or visualised. Other numeric features can also be used to calculate mean or sum

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: As the variable "postHashtags" do not have any natural order or numeric meaning, it can be stored as a non-numeric categoric variable. Data can be stored as characters (or as strings). The data can filtered easily for specific hashtags that is relevant to different posts. It can allow groupings for different themes, therefore allowing insights to latest topic or trends

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: For timestamps, a numeric continuous double variable. As time is finite and contains multiple/infinite decimal places, it can be displayed precisely all the way to the milliseconds. For example, 11.4324 seconds.

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: To represent date and time of each session, a numeric categorical data types can be used, as dates can be displayed as DD/MM/YYYY while time can be displayed as HH:MM. For such event database, it is mainly to store the session information, with no natural order.

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: To store list of nominated candidates, a character (array of string) would be ideal. This data type has no natural order. Additionally, this increases the flexibility where some participants can have different numbers of nomination. Therefore, the structure would remain organised and more accessible for further analysis.

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: The variable "preferredChannel" contains a Nominal Categorical Variable. Options like "email," "phone," or "social media" conveys the perfered mattered of communication. Therefore, it can be stored as character that has no natural order and unordered descriptions.

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: For variable "feedbackColour", an ordinal categorical data type would be ideal. For example, "warm blue" and "cool blue" conveys an indication feelings. The order of the colour temperature can be ranked based on their personal feelings about the website. This tend allows for better visualisation through graphs and analysis in finding the overall trends amongst participants

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:

time_spent: Numeric Continuous (Double). This represent exact timespent on different social media on a daily basis, and able to contain decimal places. post_frequency: Numeric Discrete (Integer). This represents the number of time the users post everyday. It can only be expressed in whole numbers. app_used : Non-numeric Categorical Nominal (Character). This represents the type of social media used by the users. Such values may include "TikTok", "Instagram", "Facebook"

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:

```
# Enter code here

library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.2     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.3     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
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:

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

print(student_info)
```

```
## $names
## [1] "Alice"    "Bob"      "Catherine"
##
## $scores
## [1] 85 92 78
##
## $passed
## [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:

```
# Enter code here
data <- c(10, 15.5, "20", TRUE)

print(typeof(data[1]))
```

```
## [1] "characters"
```

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

```
## [1] "characters"
```

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

```
## [1] "characters"
```

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

```
## [1] "characters"
```

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:

```
# Enter code here

prices <- c(20.5, 15, "25")
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:

```
# Enter code here

x <- c(5,10,15)
y <- c('apple', 'banana', 'cherry')
z <- c(x, y)
typeof(z)
```

```
## [1] "characters"
```

```
#R converts the integer to character. With implicit coercion, the vector is only able to store a single data type
```

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:

```
# Enter code here
number <- c(7,12.5,'15.7')
number <- as.numeric(number)
number_total <- sum(number)
print(number_total)
```

```
## [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:

```
# Enter code here
#As '72.5' is considered as the character, a warning message appears
grades <- c(85 ,90.5 , "72.5")
mean(grades)
```

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

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

```
grades <- c(85 ,90.5 , "72.5")
grades <- as.numeric(grades)
mean(grades)
```

```
## [1] 82.66667
```

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:

```
# Enter code here

mixed_data <- list(
  numeric = c(10, 20, 30),
  character = c("red", "green", "blue"),
  logical = c(TRUE, FALSE, TRUE)
)

print(mean(mixed_data$numeric))
```

```
## [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:

```
# Enter code here

print(student_info)
```

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

```
mean(student_info$scores)
```

```
## [1] 85
```

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:

```
# Enter code here

numeric_vector <- c(2,4,6,8,10,12,14)
print(tail(numeric_vector,1))
```

```
## [1] 14
```

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:

```
words <- c('apple', 'banana', 'cherry', 'apple')
apple_occurences <- which(words == 'apple')
print(apple_occurences)
```

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

```
# Enter code here
```

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:

```
# Enter code here

all_ages <- c(23,34,21,13,50,60)
older_than_30 <- all_ages[all_ages > 30]
print (older_than_30)
```

```
## [1] 34 50 60
```

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:

```
# Enter code here

sequence_vector <- c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20)
every_third_sequence <- sequence_vector[seq(3,length(sequence_vector), by=3)]
print(every_third_sequence)
```

```
## [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:

```
# Enter code here
nums <- c(1,2,3,4,5,6,7,8,9,10)
print(nums[c(4,5,6,7,8)])
```

```
## [1] 4 5
```

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:

```
# Enter code here
data <- c(10, NA, 15, 20)
data_NA <- which(is.na(data))
print(data_NA)
```

```
## [1] 2
```

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:

```
# Enter code here
temperature <- c(87,65,97,63,93,91,94,96,99)
hot_days <- temperature > 90
print(hot_days)
```

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

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:

```
# Enter code here

fruits <- c('apple','banana','watermelon','wintermelon')
long_names <- nchar(fruits) > 6
print (fruits[long_names])
```

```
## [1] "watermelon"  "wintermelon"
```

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:

```
# Enter code here
number_vector <- c(1,5,3,7,9,15,12,15)
divisible_by_5 <- number_vector[number_vector%%5==0]
print(divisible_by_5)
```

```
## [1]  5 15 15
```

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:

```
# Enter code here
vector1 <- c(5,2,8,4,6,3)
vector2 <- c(3,3,7,5,7,2)

comparison <- vector1 > vector2
print(comparison)
```

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