Kolowkium I.

Programowanie w języku R, 16.06.2025

Instrukcja

Zadania rozwiązywać można w dowolnej kolejności, ale proszę o zaznaczenie tego odpowiednio w kodzie (np poprzez komentarz ## Zadanie x.).

Zwracamy dwa pliki:

- skrypt źródłowy R lub notatnik rmd/qmd
- rozwiązanie wyeksportowane do formatu html/pdf.

W przypadku problemów z kompilacją do html/pdf należy zakomentować kod uniemożliwiający eksport i ponowić próbę. W ostateczności wystarczy tylko plik z kodem źródłowym.

Pliki z pracą zapisujemy w formacie `imie_nazwisko.*` (np. `piotr_szyszka.R`, `piotr_szyszka.html`).

Pracujcie nad napisaniem kodu najbardziej czytelnego jak potraficie. Pamiętajcie o formatowaniu. Komentarze w kodzie nie są obowiązkowe, ale będą doceniane.

Punktacja

Zadanie	Punkty
1	5
2	6
3	6
4	7
5	6
6	7
7	6
8	7
Razem	**50**

Zadania

1. Utwórz macierz A rozmiaru 5x5 z liczb od 1 do 25.

Następnie

- wyznacz macierz trójkątną dolną, trójkątną górną oraz przekątną macierzy,
- oblicz sumę elementów w każdym wierszu i w każdej kolumnie,
- wyświetl wektor składający się tylko z liczb parzystych tej macierzy.

2. Napisz funkcję

```
geom_mean <- function(x){
   ...
}
```

która oblicza średnią geometryczną z wektora liczb dodatnich zgodnie ze wzorem:

$$G = \sqrt[n]{\prod_{i=1}^{n} x_i}, \quad x_i > 0$$

Sprawdź wynik dla poniższych przypadków:

```
x1 <- 1:10
x2 <- c(x1, -2)
```

Wskazówka: pamiętaj o obsłudze wyjątków.

3. Napisz funkcję

```
anymiss <- function(x){
   ...
}
```

Celem funkcji jest zbadanie czy w wektorze są braki danych. Funkcja zwraca wartość logiczną (`TRUE`/`FALSE`). Jeżeli w wektorze występując jakieś braki, niech wyświetlony zostanie wektor indeksów w których są braki danych.

4. Stwórz ramki

```
klienci <- data.frame(
  id = 1:6,
  imie = c("Anna", "Jan", "Katarzyna", "Piotr", "Magda", "Tomasz"),
  miasto = c("Kraków", "Warszawa", "Gdańsk", "Poznań", "Wrocław", "Łódź"),
  kraj = c("Polska", "Polska", "Polska", "Polska", "Polska", "Polska"),
  czy_aktywny = c(TRUE, TRUE, FALSE, TRUE, FALSE, TRUE)
)

zamowienia <- data.frame(
  id_zamowienia = 101:110,
```

```
  id_klienta = c(2, 3, 4, 2, 1, 6, 3, 8, 9, 4),
  produkt = c("Laptop", "Telefon", "Monitor", "Słuchawki", "Klawiatura", "Mysz", "Drukarka",
  kwota = c(3200, 1500, 800, 450, 200, 100, 600, 1200, 900, 350),
  data = as.Date(c("2023-01-10", "2023-02-15", "2023-03-20", "2023-01-25", "2023-04-05", "20
)
```

Następnie połącz ramki ze sobą tak, aby w każdym wierszu mieć informację o zamówieniu i o kliencie. Jakiego połączenia użyjesz i dlaczego? Policz łączną kwotę zamówień na klienta (dla istniejących klientów).

5. Dla danych ze zbioru `iris` dodaj zmienne `Sepal.Area = Sepal.Length * Sepal.Width` oraz `Petal.Area = Petal.Length * Petal.Width`. Oblicz średnie powierzchni tych nowych zmiennych w podziale na grupy `Species`.

6. Dla danych ze zbioru `ToothGrowth` porównaj rozkład długości zębów (`len`) w zależności od dawki witaminy C (`dose`). Zwizualizuj wyniki w postaci wykresu pudełkowego i funkcji gęstości. Pamiętaj o skonfigurowaniu elementów wykresu (podpisy osi, tytuł itp.).

7. Dla wektora napisów

```
nazwy <- c("Kawa_arabica", "Herbata_czarna", "Yerba_mate", "Sok_pomarańczowy")
```

(a) zamień wszystkie `_` na spacje.

(b) zamień wszystkie litery na wielkie,

(c) podziel napisy na pojedyncze słowa.

8. Ustaw jądro generatora na 2025 (`set.seed(2025)`). Wygeneruj 500 obserwacji z rozkładu wykładniczego o parametrze $\lambda = 0.1$ oraz 500 z rozkładu normalnego o średniej 50 i std. 10. Połącz obie próbki w jeden wektor.

W zależności od kwantyli (funkcja `quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1))`) podziel:

```
- "Niski" (x < Q1),

- "Średni" (Q1   x < Q2),

- "Wysoki" (Q2   x < Q3),

- "Bardzo wysoki" (x   Q3).
```

Wyświetl tabelę częstości dla przedziałów.