Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 3

з дисципліни

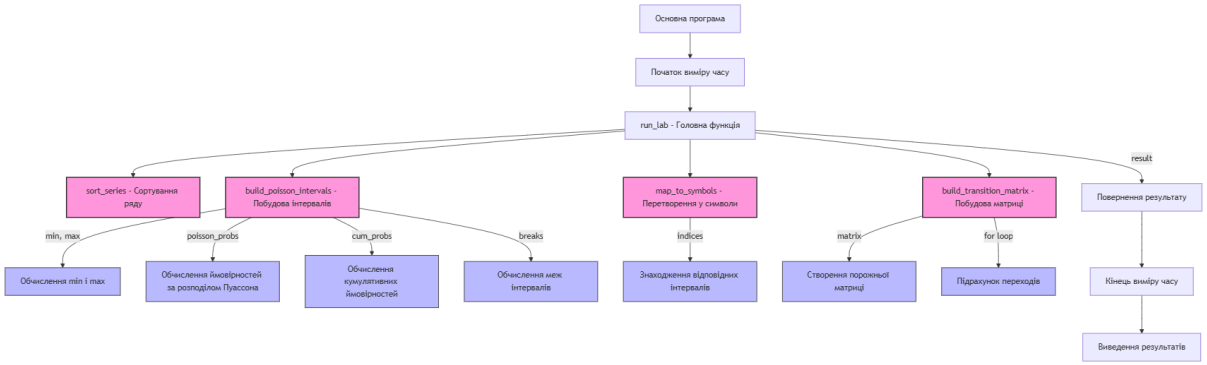
«Мультипарадигменне програмування»

ВИКОНАВ:
студент III курсу ФІОТ
групи ІО-23
Бичкар Н. В.
Залікова № 2302

ПЕРЕВІРИВ:
ас. Очеретяний О. К.

Київ – 2025

Функціональна схема (взаємозв'язку функцій):



Тексти визначень функцій:
Тексти визначень функцій у програмі
1. Функція сортування числового ряду

```
# Функція для сортування числового ряду
sort_series <- function(series) {
  sort(series)
}
```

Призначення: Сортує вхідний числовий ряд від найменшого до найбільшого значення.
Вхідні параметри:

- series - вектор чисельних значень вхідного ряду

Вихідні дані:

- Відсортований вектор чисельних значень

Алгоритм роботи:

1. Використовує вбудовану функцію R sort() для сортування вектора
2. Функція побудови інтервалів за розподілом Пуассона

```
# Побудова інтервалів на основі пуассонівського розподілу
build_poisson_intervals <- function(series, alphabet_size, lambda = 1) {
  min_val <- min(series)
  max_val <- max(series)

  # Створюємо вектор ймовірностей за розподілом Пуассона
  x <- 0:(alphabet_size-1)
  poisson_probs <- dpois(x, lambda)
  poisson_probs <- poisson_probs / sum(poisson_probs)  # Нормалізація ймовірностей

  # Обчислюємо кумулятивні ймовірності
  cum_probs <- cumsum(poisson_probs)
```

```
  # Створюємо межі інтервалів на основі кумулятивних ймовірностей
  range <- max_val - min_val
  breaks <- min_val + cum_probs * range
  breaks <- c(min_val, breaks)

  return(breaks)
}
```

Призначення: Створює межі інтервалів відповідно до розподілу Пуассона для подальшого перетворення числового ряду в лінгвістичний.
Вхідні параметри:

- series - вектор чисельних значень (зазвичай відсортований)
- alphabet_size - розмір алфавіту (кількість символів/інтервалів)
- lambda - параметр розподілу Пуассона (за замовчуванням = 1)

Вихідні дані:

- Вектор меж інтервалів за розподілом Пуассона

Алгоритм роботи:

1. Знаходить мінімальне та максимальне значення у вхідному ряді
2. Створює вектор ймовірностей за розподілом Пуассона для значень від 0 до (alphabet_size-1)
3. Нормалізує ймовірності, щоб їх сума дорівнювала 1
4. Обчислює кумулятивні ймовірності
5. Створює межі інтервалів на основі кумулятивних ймовірностей, розподіляючи їх на діапазоні від min до max
6. Додає мінімальне значення як першу межу і повертає вектор меж

3. Функція перетворення чисел у символи

```
# Перетворення чисел у відповідні символи алфавіту
map_to_symbols <- function(series, breaks, alphabet) {
  indices <- findInterval(series, breaks, rightmost.closed = TRUE)
  indices[indices == 0] <- 1
  indices[indices > length(alphabet)] <- length(alphabet)
  return(alphabet[indices])
}
```

Призначення: Перетворює числові значення у відповідні символи алфавіту на основі меж інтервалів.
Вхідні параметри:

- series - вхідний числовий ряд
- breaks - вектор меж інтервалів
- alphabet - вектор символів алфавіту

Вихідні дані:

- Вектор символів (лінгвістичний ряд)

Алгоритм роботи:

1. Використовує функцію findInterval() для визначення, до якого інтервалу належить кожне число
2. Коригує індекси, які дорівнюють 0 (встановлює їх як 1)

3. Коригує індекси, які більші за розмір алфавіту (встановлює їх як максимальний індекс)
4. Перетворює індекси у відповідні символи алфавіту

4. Функція побудови матриці передування

```
# Побудова матриці передування символів
build_transition_matrix <- function(symbols, alphabet) {
  n <- length(alphabet)
  matrix <- matrix(0, nrow = n, ncol = n, dimnames = list(alphabet, alphabet))

  for (i in 1:(length(symbols) - 1)) {
    from <- symbols[i]
    to <- symbols[i + 1]
    matrix[from, to] <- matrix[from, to] + 1
  }

  return(matrix)
}
```

Призначення: Будує матрицю передування символів на основі лінгвістичного ряду.
Вхідні параметри:

- symbols - вектор символів (лінгвістичний ряд)
- alphabet - вектор символів алфавіту

Вихідні дані:

- Матриця передування, де елемент [i,j] містить кількість переходів від символу і до символу j

Алгоритм роботи:

1. Створює матрицю нулів розміром n×n, де n - розмір алфавіту
2. Встановлює назви рядків і стовпців матриці як символи алфавіту
3. Для кожної пари послідовних символів у лінгвістичному ряді: a. Визначає символ, з якого відбувається перехід (from) b. Визначає символ, до якого відбувається перехід (to) c. Збільшує відповідний елемент матриці на 1
4. Повертає заповнену матрицю

5. Головна функція програми

```
# Головна функція
run_lab <- function(series, alphabet, lambda = 1) {
  alphabet_size <- length(alphabet)
  sorted <- sort_series(series)

  # Використовуємо розподіл Пуассона для створення інтервалів
  breaks <- build_poisson_intervals(sorted, alphabet_size, lambda)

  symbols <- map_to_symbols(series, breaks, alphabet)
  transition_matrix <- build_transition_matrix(symbols, alphabet)

  list(
    series = series,
    sorted = sorted,
    breaks = breaks,
```

```
    symbols = symbols,
    matrix = transition_matrix
  )
}
```

Призначення: Координує процес перетворення числового ряду в лінгвістичний та побудови матриці передування.
Вхідні параметри:

- series - вхідний числовий ряд
- alphabet - вектор символів алфавіту
- lambda - параметр розподілу Пуассона (за замовчуванням = 1)

Вихідні дані:

- Список (list), що містить:
    - series - вхідний числовий ряд
    - sorted - відсортований числовий ряд
    - breaks - межі інтервалів
    - symbols - лінгвістичний ряд
    - matrix - матриця передування

Алгоритм роботи:

1. Визначає розмір алфавіту
2. Сортує вхідний числовий ряд
3. Створює межі інтервалів відповідно до розподілу Пуассона
4. Перетворює числовий ряд у лінгвістичний
5. Будує матрицю передування
6. Повертає список з результатами

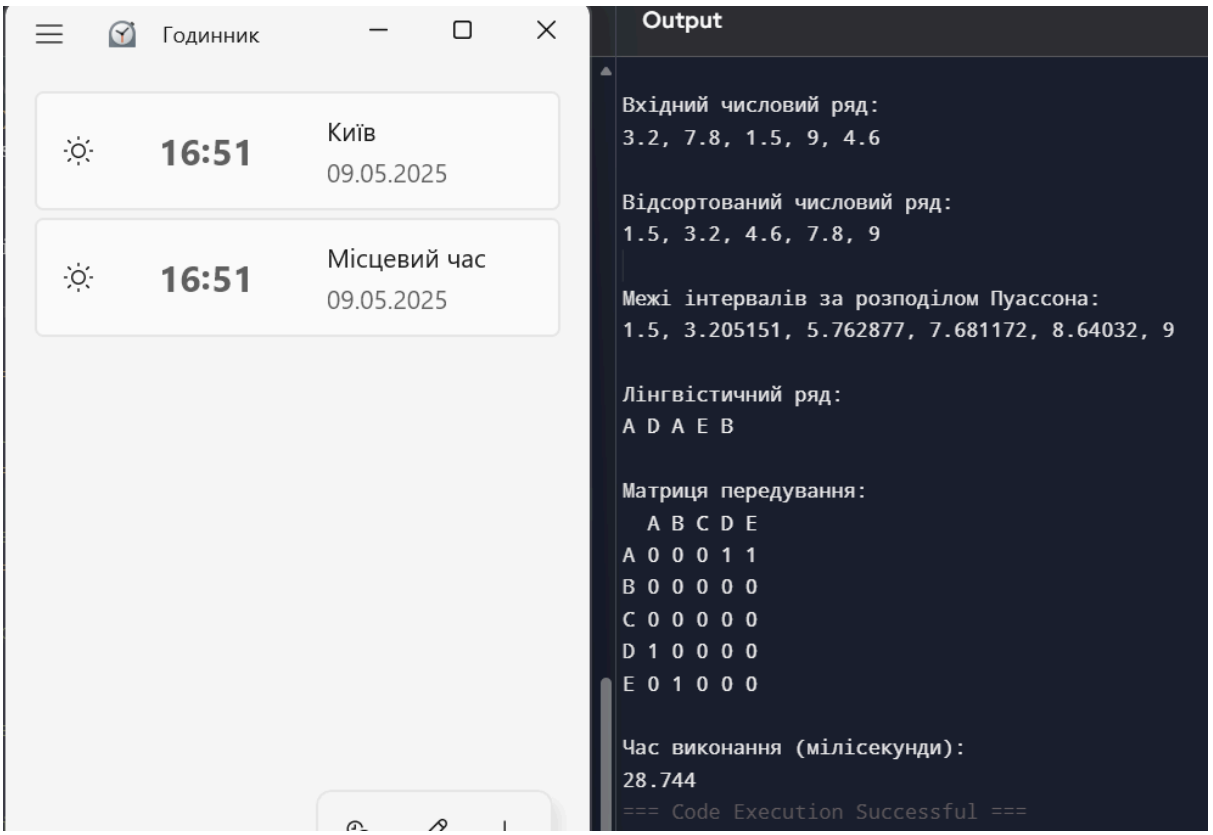
Лістинг:

```
# === 1. Задаємо числовий ряд ===
series <- c(3.2, 7.8, 1.5, 9.0, 4.6)

# === 2. Визначення функцій ===

# Функція для сортування числового ряду
sort_series <- function(series) {
  sort(series)
}

# Побудова інтервалів на основі пуассонівського розподілу
build_poisson_intervals <- function(series, alphabet_size, lambda = 1) {
  min_val <- min(series)
  max_val <- max(series)

  # Створюємо вектор ймовірностей за розподілом Пуассона
  x <- 0:(alphabet_size-1)
  poisson_probs <- dpois(x, lambda)
  poisson_probs <- poisson_probs / sum(poisson_probs)  # Нормалізація ймовірностей

  # Обчислюємо кумулятивні ймовірності
```

```
  cum_probs <- cumsum(poisson_probs)

  # Створюємо межі інтервалів на основі кумулятивних ймовірностей
  range <- max_val - min_val
  breaks <- min_val + cum_probs * range
  breaks <- c(min_val, breaks)

  return(breaks)
}

# Перетворення чисел у відповідні символи алфавіту
map_to_symbols <- function(series, breaks, alphabet) {
  indices <- findInterval(series, breaks, rightmost.closed = TRUE)
  indices[indices == 0] <- 1
  indices[indices > length(alphabet)] <- length(alphabet)
  return(alphabet[indices])
}

# Побудова матриці передування символів
build_transition_matrix <- function(symbols, alphabet) {
  n <- length(alphabet)
  matrix <- matrix(0, nrow = n, ncol = n, dimnames = list(alphabet, alphabet))

  for (i in 1:(length(symbols) - 1)) {
    from <- symbols[i]
    to <- symbols[i + 1]
    matrix[from, to] <- matrix[from, to] + 1
  }

  return(matrix)
}

# Головна функція
run_lab <- function(series, alphabet, lambda = 1) {
  alphabet_size <- length(alphabet)
  sorted <- sort_series(series)

  # Використовуємо розподіл Пуассона для створення інтервалів
  breaks <- build_poisson_intervals(sorted, alphabet_size, lambda)

  symbols <- map_to_symbols(series, breaks, alphabet)
  transition_matrix <- build_transition_matrix(symbols, alphabet)

  list(
    series = series,
    sorted = sorted,
    breaks = breaks,
    symbols = symbols,
```

```
    matrix = transition_matrix
  )
}

# === 3. Визначення алфавіту ===
alphabet <- c("A", "B", "C", "D", "E")

# === 4. Запуск алгоритму з вимірюванням часу виконання ===
start_time <- Sys.time()
result <- run_lab(series, alphabet, lambda = 1.5)  # Використовуємо параметр lambda =
1.5 для прикладу
end_time <- Sys.time()
execution_time <- end_time - start_time

# === 5. Вивід результатів ===
cat("\nВхідний числовий ряд:\n")
cat(result$series, sep = ", ")

cat("\n\nВідсортований числовий ряд:\n")
cat(result$sorted, sep = ", ")

cat("\n\nМежі інтервалів за розподілом Пуассона:\n")
cat(result$breaks, sep = ", ")

cat("\n\nЛінгвістичний ряд:\n")
cat(result$symbols, sep = " ")

cat("\n\nМатриця передування:\n")
print(result$matrix)

cat("\nЧас виконання (мілісекунди):\n")
cat(round(as.numeric(execution_time, units = "secs") * 1000, 3))
```

Результати:


Годинник
16:51
Київ
09.05.2025
16:51
Місцевий час
09.05.2025
Output
Вхідний числовий ряд:
3.2, 7.8, 1.5, 9, 4.6
Відсортований числовий ряд:
1.5, 3.2, 4.6, 7.8, 9
Межі інтервалів за розподілом Пуассона:
1.5, 3.205151, 5.762877, 7.681172, 8.64032, 9
Лінгвістичний ряд:
A D A E B
Матриця передування:
A B C D E
A 0 0 0 1 1
B 0 0 0 0 0
C 0 0 0 0 0
D 1 0 0 0 0
E 0 1 0 0 0
Час виконання (мілісекунди):
28.744
=== Code Execution Successful ===