MatrixMultiplier – Opis programu i omówienie kodu

Autor: Adrian Kasprzak

Cel programu

Program tworzy dwie macierze 5x5 z liczbami losowymi z zakresu 0–10, następnie mnoży je zgodnie z zasadami algebry liniowej. Jeżeli w którymkolwiek wierszu pierwszej macierzy znajduje się zero, ten wiersz zostaje pominięty w operacji mnożenia (nie jest liczony do wyniku).

Struktura projektu

- **MatrixMultiplier.java** – plik główny, klasa z metodą `main()`
- **MatrixHelpers.java** – klasa pomocnicza z metodami do drukowania macierzy i sprawdzania obecności zera w wierszu

Opis działania krok po kroku

1. **Inicjalizacja macierzy i generatora liczb losowych:**
    - Tworzone są trzy dwuwymiarowe tablice typu `int` o wymiarach 5x5:
        - `matrixA` – pierwsza macierz wejściowa
        - `matrixB` – druga macierz wejściowa
        - `result` – macierz wynikowa
    - Używany jest obiekt klasy `Random` do losowania wartości
2. **Wypełnianie macierzy liczbami losowymi:**
    - Za pomocą zagnieżdżonych pętli `for` oba wejściowe zbiory danych (matrixA, matrixB) są wypełniane liczbami z zakresu 0–10.

3. **Wyświetlanie macierzy:**
    - Używana jest metoda pomocnicza `printMatrix` z klasy `MatrixHelpers` do estetycznego wydruku obu macierzy wejściowych na konsoli.
4. **Mnożenie macierzy z pomijaniem wierszy zawierających zero:**
    - Iteracja po wierszach pierwszej macierzy (pętla po `i`).
    - Przed mnożeniem sprawdzane jest, czy dany wiersz zawiera zero (metoda `containsZero`).
    - Jeśli wiersz zawiera zero, wyświetlana jest informacja i ten wiersz jest pomijany w dalszych obliczeniach (`continue`).
    - Jeśli nie zawiera zera, następuje klasyczne mnożenie wiersza przez kolumnę drugiej macierzy – suma iloczynów elementów (`sum += matrixA[i][k] * matrixB[k][j]`).
    - Wyniki trafiają do macierzy `result`.
5. **Wyświetlanie wyniku:**
    - Macierz wynikowa jest drukowana na konsoli.

Opis użytych klas, metod i typów danych

Klasa `MatrixMultiplier`

- Zawiera punkt startowy programu: metodę `main()`.
- Klasa dziedziczy po `MatrixHelpers` (lub korzysta z jej metod statycznych).

Metoda `main(String[] args)`

- Uruchamia całą logikę programu: inicjuje macierze, wywołuje funkcje pomocnicze i zarządza przepływem sterowania.

Klasa `MatrixHelpers`

- Zawiera metody:

- `containsZero(int[] row)` – zwraca `true` jeśli wiersz zawiera zero
- `printMatrix(int[][] matrix)` – wypisuje macierz w formie czytelnej tabeli

Typy danych

- `int[][]` – tablica dwuwymiarowa liczb całkowitych, reprezentująca macierz 5x5
- `Random` – klasa z biblioteki Java, używana do generowania liczb losowych

Użyte pętle

- Dwie zagnieżdżone pętle `for` do wypełniania macierzy losowymi wartościami
- Trzy zagnieżdżone pętle `for` w trakcie mnożenia macierzy (po wierszach, kolumnach i sumowanych elementach)
- Pętla `for-each` w metodzie pomocniczej do drukowania i sprawdzania wierszy

Warunki sterujące

- Instrukcja warunkowa `if (containsZero(matrixA[i]))` – pozwala pominąć niepotrzebne obliczenia dla wierszy, w których i tak wyniki będą nieistotne (albo zerowe)
- Instrukcja `continue` – przechodzi od razu do następnego wiersza w pętli

Przykładowy fragment kodu

```
for (int i = 0; i < SIZE; i++) {
    if (containsZero(matrixA[i])) {
        System.out.println("Wiersz " + i + " w macierzy A zawiera zero – pomijam w
mnożeniu.");
        continue;
    }
    for (int j = 0; j < SIZE; j++) {
        int sum = 0;
        for (int k = 0; k < SIZE; k++) {
            sum += matrixA[i][k] * matrixB[k][j];
        }
        result[i][j] = sum;
```

```
    }
}
```

Sposób budowania i uruchamiania programu

1. Kompilacja

Skompiluj wszystkie pliki razem, będąc w folderze z plikami `.java`:

```
javac MatrixMultiplier.java MatrixMultiplier.java
```

Powstaną pliki `.class` dla każdej klasy/interfejsu.

2. Uruchomienie

Program uruchamiasz komendą:

```
java MatrixMultiplier
```

Najważniejsze zalety rozwiązania

- Optymalizacja: nie są wykonywane niepotrzebne obliczenia dla wierszy, które i tak zawierają zero
- Jasna, czytelna struktura kodu (podział na klasy i metody)
- Prosta rozbudowa i łatwość testowania

Podsumowanie

Program demonstruje pracę na macierzach i operacje zgodne z algebrą liniową, pokazując zarówno umiejętność korzystania z zagnieżdżonych pętli i warunków, jak i

poprawny podział kodu na klasy oraz metody wspierające czytelność i ponowne użycie funkcji.