ZADANIE #33 – PRZESUWANIE ELEMENTÓW W PRAWO O JEDEN

Napisz program, w którym zdefiniujesz tablicę z kilkoma wartościami typu `int`. Korzystając z pętli, przesuń wszystkie elementy tej tablicy w prawo o `1`. Oznacza to, że pierwszy element powinien zostać umieszczony na drugim miejscu w tablicy, drugi element na trzecim, trzeci na czwartym i tak dalej. Ostatni element w tym przypadku jest szczególnym przypadkiem – powinien on stać się pierwszym elementem tablicy. Po przesunięciu elementów, wypisz zawartość tablicy na ekran w jednej linii.

Przykładowe dane i oczekiwany wynik uruchomienia programu

Jeżeli zdefiniowana tablica będzie składała się z elementów `100`, `5`, `0`, `-20`, `15`, to program powinien wypisać na ekran:

```
15 100 5 0 -20
```

Rozwiązanie

W rozwiązaniu do tego zadania musimy w odpowiedni sposób przejść przez zdefiniowaną przez nas tablicę i przesunąć w prawo każdy z elementów, poza ostatnim, ponieważ, wedle opisu zadania, powinien on stać się pierwszym elementem tablicy.

Plik: 08_tablice/PrzesuwanieElementowWPrawoOJeden.java

```
public class PrzesuwanieElementowWPrawoOJeden {
  public static void main(String[] args) {
    int[] liczby = { 100, 5, 0, -20, 15 }; // 1

    int ostatniElement = liczby[liczby.length - 1]; // 2

    for (int i = liczby.length - 1; i > 0 ; i--) { // 3
      liczby[i] = liczby[i - 1]; // 4
    }

    liczby[0] = ostatniElement; // 5

    for (int liczba : liczby) { // 6
      System.out.print(liczba + " ");
    }
  }
}
```

W linii `(1)` definiujemy tablicę `liczby`, której elementy będziemy przesuwać.

Następnie, w linii `(2)`, zapisujemy w pomocniczej zmiennej `ostatniElement` wartość ostatniego elementu tablicy `liczby`. Za chwilę wyjaśnię, dlaczego ta zmienna jest nam potrzebna. Zwróć uwagę, że indeks ostatniego elementu to długość tablicy pomniejszona o jeden, ponieważ w języku Java numeracja elementów tablic zaczyna się od `0`, a nie `1`. Gdybyśmy nie odjęli `1` od długości tablicy, to wyszlibyśmy poza zakres jej elementów.

Główną częścią programu jest pętla `for` rozpoczynająca się w linii `(3)`. Jej zadaniem jest przejście przez tablicę od końca do początku, z pominięciem pierwszego elementu tablicy (zapewnia to warunek pętli `i > 0` zamiast `i >= 0`). W każdym obiegu pętli, w linii `(4)`, przypisujemy do elementu tablicy wartość poprzedzającego go elementu. W ten sposób, w pierwszym obiegu pętli przypiszemy do ostatniego elementu wartość przedostatniego elementu tablicy. W drugim obiegu do przedostatniego elementu przypiszemy wartość przed-przedostatniego elementu i tak dalej. Spójrz na poniższą reprezentację działania pętli `for` z linii `(3)`:

Przed rozpoczęciem pętli

Indeks elementu	0	1	2	3	4
Wartość elementu	100	5	0	-20	15

Po pierwszym obiegu pętli (wytłuszczona wartość to ta, która została w tym obiegu przesunięta)

Indeks elementu	0	1	2	3	4
Wartość elementu	100	5	0	-20	**-20**

Po drugim obiegu pętli

Indeks elementu	0	1	2	3	4
Wartość elementu	100	5	0	**0**	-20

Po trzecim obiegu pętli

Indeks elementu	0	1	2	3	4
Wartość elementu	100	5	**5**	0	-20

Po czwartym obiegu pętli

Indeks elementu	0	1	2	3	4
Wartość elementu	100	**100**	5	0	-20

Po zakończeniu pętli

Indeks elementu	0	1	2	3	4
Wartość elementu	100	100	5	0	-20

Po wstawieniu wartości `ostatniElement` *na początek tablicy*

Indeks elementu	0	1	2	3	4
Wartość elementu	15	100	5	0	-20

Zauważ, że po każdym zaprezentowanym obiegu pętli, kolejna wartość „wędruje” na miejsce swojego „prawego” sąsiada. W rezultacie, wszystkie elementy tablicy przesuną się w prawo o jeden, poza ostatnim elementem, którego wartość utraciliśmy. Nie ma jej już w tablicy `liczby`, ponieważ została nadpisana przedostatnią wartością tablicy, która zajęła jej miejsce, co widać w tabeli powyżej zatytułowanej „Po zakończeniu pętli”. Dlatego przed pętlą, w linii `(2)`, zapisaliśmy w zmiennej `ostatnieElement` wartość ostatniego element tablicy `liczby` – możemy teraz tę wartość umieścić na początku tablicy, co czynimy w linii `(5)`, tym samym dokańczając przesuwanie elementów.

Dlaczego warunek pętli to `i > 0` zamiast `i >= 0` ? Zauważ, że w ciele pętli odnosimy się do elementu poprzedzającego element, którego indeks to wartość zmiennej `i`. Element o indeksie `0` nie ma poprzedniego elementu, więc wyszlibyśmy poza zakres tablicy próbując odnieść się do nieistniejącego elementu o indeksie `-1`.

Na końcu programu pozostaje nam wypisać elementy tablicy na ekran, którą to operację wykonujemy za pomocą pętli `for`-each w linii `(6)`.

Przykładowe uruchomienie programu

Gdy zdefiniowana tablica składa się z elementów `100`, `5`, `0`, `-20`, `15`, to program wypisuje na ekran:

```
15 100 5 0 -20
```