Wzorce Projektowe, lab. 12

Temat: Praktyczne zastosowanie wzorca **Strategy**.

Twoje zadanie to implementacja struktury, której elementy będą liczbami całkowitymi oraz będą zawsze posortowane. Będziemy jednak chcieli móc skorzystać z kilku algorytmów sortowania i do tego można wykorzystać wzorzec **Strategy**.

1. Utwórz interfejs `StrategiaSort` z jedną metodą: `void sortuj(int[] tab)`.
2. Zaimplementuj ten interfejs w dwóch klasach, które będą określały różne algorytmy sortowania. Możesz wybrać dowolne algorytmy, np. sortowanie bąbelkowe i quicksort – klasy powinny wtedy nazywać się `StrategiaSortBabelkowe` i `StrategiaSortQuicksort`. Informacje o tym, jak zaimplementować algorytmy sortowania, są dostępne w internecie.
3. Przejdźmy do tworzenia kontekstu. Utwórz klasę `PosortowaneLiczby`, w której będziesz przechowywał prywatny obiekt typu `StrategiaSort` (przekaż go w konstruktorze) i jakiś zbiór liczb. Możesz do tego celu wykorzystać np. `ArrayList<Integer>` lub zrobić to po swojemu, o ile program nie będzie kompletnie nieefektywny (czyli np. tworzenie nowej tablicy i przekopiowywanie wszystkich wartości za każdym razem, gdy użytkownik dodaje nową liczbę). W klasie mają zostać zaimplementowane metody:
   - `public void addNumber(int i)`
   - `public void addArray(int[] tab)`
   - `public int[] getArray()`
   - `public String toString()`

   Zastanów się, kiedy musisz sortować liczby i sortuj poprzez przekazanie sterowania obiektowi klasy `StrategiaSort`.
4. W funkcji *main()* utwórz nowy obiekt klasy `PosortowaneLiczby` z pierwszą strategią i wypełnij 5 liczbami (możesz wpisać je „na sztywno”). Następnie wypisz cały obiekt na konsoli (program skorzysta wtedy z metody `public String toString()`). Po wypisaniu dodaj do niego kolejne liczby (skorzystaj z obydwu funkcji do dodawania) i ponownie wypisz wszystko. Później utwórz nowy obiekt `PosortowaneLiczby`, ale tym razem z drugą zaimplementowaną przez Ciebie strategią i wykonaj te same kroki, co dla poprzedniego obiektu.