Lista zadań nr 5: ***Sortowanie (algorytmy proste)***

Zadanie 1.

Implementując każdy w omawianych na wykładzie *prosty algorytm sortowania* zbadaj (na stosownych, samodzielnie zdefiniowanych zbiorach danych testowych) liczbę porównań i przesunięć (przepisań) wykonanych w każdym z algorytmów. Sformułuj stosowne wnioski.

Zadanie 2.

Zaimplementuj każdy z omawianych na wykładzie *prosty algorytm sortowania* na zbiorze obiektów danego (zaproponowanego przez siebie) typu tak, by możliwe było uzyskiwanie kilku alternatywnych porządków, wyznaczanych przez wartości różnych atrybutów (pól).

Zadanie 3.

Zaproponuj implementację realizującą wielość alternatywnych porządków zbioru danych z zadania 2. z użyciem ogólnych mechanizmów omawianych na wykładzie (w tym: comparatora). Porównaj to rozwiązanie z rozwiązaniem zadania 2.

Zadanie 4. – dla hobbystów

Zaproponuj ogólną, ulepszoną w stosunku do wersji podstawowej, implementację sortowania bąbelkowego z *przechodzeniem* ciągu, na przemian, w obu kierunkach (ShakerSort) tak, by po każdym przejściu (w wyniku którego element maksymalny/minimalny trafia na swoje docelowe miejsce) skracać zakres *przechodzenia pozostałego (nieuporządkowanego) fragmentu* ciągu. Wprowadź zapamiętywanie miejsca ostatniego przestawienia elementów, by przyspieszyć sortowanie.