Algorytmy i struktury danych – sortowanie

Implementacja algorytmów sortowania

Napisz cztery funkcje sortujące, które implementują następujące algorytmy:

- sortowanie bąbelkowe (ang. *bubble sort*),
- sortowanie przez wybieranie (ang. *selection sort*),
- sortowanie przez scalanie (ang. *merge sort*),
- sortowanie szybkie (ang. *quick sort*).

Każda z funkcji powinna przyjmować jako argument listę i zwracać listę posortowaną, np.:

```
>>> bubble_sort([3,5,1])
[1,3,5]
```

Porównanie algorytmów sortowania

Jako dane do sortowania wykorzystaj plik `pan-tadeusz.txt`, zawierający słowa oddzielone białymi znakami. Dla każdej z funkcji sortujących:

- sprawdź czy funkcja poprawnie sortuje słowa wczytywane z pliku,
- zmierz czas sortowania list zawierających *n* pierwszych słów wczytanych z pliku (np. *n* = 1000, 2000, ..., 10000),
- wygeneruj wykres zależności czasu sortowania od długości listy.

Zwróć uwagę by mierzyć wyłącznie czas sortowania, pomijając wczytywanie danych lub wyświetlanie wyników. Informację o funkcjach i bibliotekach, które możesz wykorzystać do pomiaru czasu i generowania wykresów znajdziesz w pliku `AISDI-Wskazowki.pdf`.

Wyniki

Rezultatem powinny być:

- kod źródłowy z zaimplementowanymi funkcjami sortującymi,
- kod źródłowy przeprowadzający komplet pomiarów wydajności i generujący pliki PNG z wykresami,
- wygenerowane pliki PNG z wykresami.

Ocena

Zadanie oceniane jest w skali 0-6 pkt.