Projekt 3 - wersja podstawowa

Temat: Sortowanie

Celem tego projektu jest porównanie czasu działania algorytmów sortujących.

Informacje organizacyjne:

1. Rekordy do posortowania są typu `T = unsigned short`:
2. Recordy do posortowania mają być umieszczone w `std::vector`
3. Do implementacji musi być dołączone:
    - minimum dziesięć zbiorów testowych umieszczonych w plikach o nazwach `set_XX.txt`. Mile widziane jest:
        - zróżnicowanie wielkości zbiorów testowych
        - demonstracja przypadku pesymistycznego/optymistycznego
    - wyniki działania programu dla zbiorów testowych umieszczonych w plikach o nazwach `set_XX_output.txt`
    - pliki z implementacją algorytmów i funkcji pomocniczych
    - program demonstrujący (patrz wymgania), kod programu testujacego powinien być w pliku o nazwie: `projekt_3_<inicjaly_TL>_<inicjaly_>_demo.cpp`

Wymagania podstawowe (muszą być spełnione)

1. Zaimplementować dwa algorytmy sortujące.
2. Napisać program demonstracyjny, który
    - wczytuje z pliku zbiór do posortowania; nazwę pliku podaje użytkownik po uruchomieniu programu
    - sortuje go za pomocą każdego algorytmu
    - wyświetla na ekranie czas jaki zajęło sortowanie dla każdego z algorytmów
    - zapisuje posortowany zbiór do pliku wraz z czasami sortowania i nazwami algorytmów
3. Dodatkowo, program demonstrujący umożliwia wygenerowanie losowego zbióru rekordów. W tym wypadku dodatkowym wyjściem programu jest plik zawierający wygenerowany zbiór.
4. Sprawozdanie ma zawierać:
    - nazwy zaimplementowanych algorytmów
    - opis procedury testowej
    - opis zestawów testowych np.: Zestaw XX jest losowy/posortowany/prawie posortowany/posortowany odwrotnie, więc powoduje "takie a takie zachowanie się" algorytmu AAA
    - analizę uzyskanych wyników

Wymagania dodatkowe (mogą być spełnione)

1. Zaimplementowane są przynajmniej 4 algorytmy sortujące
2. Dla tych algorytmów których to dotyczy: przetestowane jest zachowanie przy różnych wyborach strategii podziału tablicy.
3. Zbadać, który z zaimplementowanych algorytmów jest szybszy dla tablic o małych rozmiarach. Znaczenie zwrotów ,,mały rozmiar'' i ,,szybszy algorytm'' określić empirycznie, badając zachowanie zaimplementowanych algorytmów.
4. W oparciu o punkt poprzedni, zaimplementować funkcję sortującą, która w zależności od rozmiaru tablicy wybiera ,,szybszy algorytm'' sortujący. W sprawozdaniu dokładnie opisać użyty algorytm i przesłanki stojące za takim a nie innym rozwiązaniem.