Kraków 11.10.2013

Algorytmy i struktury danych I

Szybkie algorytmy do sortowania

Zaimplementuj jeden z szybkich algorytmów do sortowania:

- Quick sort
- Merge sort
- Heap sort

Dokonaj testów działania programu na zbiorach o następującej ilości elementów: 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000. Generuj dane wykorzytując programy testowe przygotowane na stronie przedmiotu.

Dla każdej liczności zbioru, dokonaj następujących testów:

1. t_{po} - czas sortowania zbioru posortowanego
2. t_{od} - czas sortowania zbioru posortowanego odwrotnie
3. t_{pp} - czas sortowania zbioru posortowanego z losowym elementem na początku
4. t_{pk} - czas sortowania zbioru posortowanego z losowym elementem na końcu
5. t_{np} - czas sortowania zbioru z losowym rozkładem elementów

Analizując czas wykonania programu oceń jaka jest złożoność obliczeniowa

- **optymistyczna:** dla zbiorów uporządkowanych (z niewielką ilością elementów nie na swoich miejscach) - na podstawie czasów t_{po}, t_{pp}, t_{pk}
- **typowa:** dla zbiorów o losowym rozkładzie elementów - na podstawie czasu t_{np}
- **pesymistyczna:** dla zbiorów posortowanych odwrotnie - na podstawie czasu t_{od}

1 Zaliczenie zestawu

- Umieszczenie w systemie PEGAZ:
  - 1 plik z kodem programu: program main sterujący procedurą wczytywania i wypisywania danych, oraz funkcja sortująca, można wykorzystać kod programu *wzorcowka.cpp*.
  - 1 przykladowy plik inputowy wykorzystany do testowania programow, w formacie jak opisany w tekscie zadania, Do generacji można użyć kod *generatorTest.cpp* + dodatkowe modyfikacje (opisane w tekscie zadania).
  - 1 plik w formacie .pdf zawierający krótki opis algorytmu oraz wyniki testów czasu działania programu.
- Ustna odpowiedź: prezentacja działającego kodu, umiejętność przedstawienia algorytmu sortującego oraz otrzymanych wynikw z testu czasu dzialania algorytmu.