Politechnika Śląska

Wydział Informatyki, Elektroniki i Informatyki

Programowanie Komputerów 3

Biblioteka sortująca

Autor	Dawid Furs
Prowadzący	dr inż. Piotr Pecka
Rok akademicki	2020/2021
Kierunek	informatyka
Rodzaj studiów	SSI
Semestr	1
Termin laboratorium	czwartek, 13:00-14:30
Sekcja	62
Termin oddania sprawozdania	2020-11-12

1. Treść zadania

   13. Napisać bibliotekę sortującą obiekty dowolnej klasy wg. Określonego pola typu np. double lub string lub innego dla, którego programista zdefinuje odpowiedni operator. Proponowane algorytmy to qsort, poste wstawianie, proste wybieranie. Przeprowadzić benchmark dla dużych, średnich i małych rozmiarów danych.

2. Analiza zadania

   W zadaniu należy zająć się tworzeniem odpowiednich class sortujących, które umożliwią za pomocą odpowiednich algorytmów posortowanie danych z pliku.

3. Algorytmy które będę stosować to wszystkie podane wyżej czyli qsort, proste wstawianie, proste wybieranie oraz sortowanie bąbelkowe.

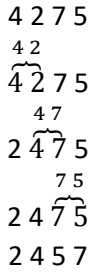
4. Specyfikacja zewnętrzna

   Program jest uruchamiany za pomocą plik typu Visual Studio Solution, po uruchomieniu programu użytkownik może wybrać w konsoli odpowiednie argumenty sortujące, sprawdzić ile zachodzi dla nich kombinacji i porównać je odpowiednio.

5. Ogólna struktura programu
   W funkcji głównej zadeklarowane są odpowiednie zmienne które przechowują dane, które następnie można użyć w wywołaniach metod poszczególnych klas.
   W programie znajduje się klasa ReadFile mająca metody umożliwiające zliczanie wyrazów z pliku, wczytywanie ich do dynamicznej tablicy, zapisywanie danych z powrotem do innego pliku tekstowego w celu umożliwienia sprawdzenia czy dane zostały posortowane, znajduje się też tam metoda która wczytuje losowe liczby do pliku, które pozwalają utworzyć plik o większej ilości danych, jest to klasa od której zachodzi najwięcej dziedziczenia. Klasa Bombelkowe, Szybkie, Wstawianie i Wybieranie te cztery klasy są odpowiednikami odpowiednich algorytmów sortujących: sortowanie bąbelkowe, qsort, poste wstawianie, proste wybieranie. Powstała również klasa Punkt pobiera ona dane z pliku, które są rozdzielane spacjami, a dane powinno się wpisywać: x y x y… w klasie Punkt te dane są obliczane i zamieniane na wektory które w dalszej części są segregowane i zapisywane w pliku tekstowym.

   Class Bombelkowe
   W tej klasie został opisany algorytm sortowania bąbelkowego, algorytm prosty w implementacji o złożoności O(n^2), sprawia to, że przy większym zbiorze danych algorytm ten może mieć poważne problemy. Sortowanie te polega na przemieszczaniu się danych ku prawej stronie i układaniu w odpowiednim szyku posortowanych danych. Algorytm ten porównuje dwie wartości do siebie i w zależności od tego na jakie dwie wartości są porównywane to większa z nich przechodzi na dalszą pozycję w prawo.

Dla przykładu:



Przy tak małym zbiorze udało nam się posortować liczby za pierwszy razem lecz w innym wypadku mogła by zostać posortowana tylko największa liczba, wtedy resztę tablicy sortowalibyśmy w ten sam sposób, ale była by ona pomniejszona o n-1, w taki sposób sortowalibyśmy tablicę tak długo, aż wszystkie elementy znalazły by się na swoim miejscu

Class Szybkie
Tutaj został opisany algorytm quicksort czyli sortowanie szybkie, jeden z najszybciej sortujących algorytmów o złożoności O(n log n), choć w sytuacji pesymistycznej może osiągnąć O(n^2), algorytm ciężki w implementacji w moim programie użyłem rekurencji do jego stworzenia, algorytm polega na technice „dziel i zwyciężaj” oznacz to że z tablicy wybiera punkt na podstawie którego dzieli tablicę na dwie części, po czym od liczby jaka znajduje się w danym punkcie układa wartości mniejsze na lewo od tego punktu, a większe na prawo, następnie z tych dwóch tablic tworzymy kolejne podtablice i wykonujemy schemat ponownie aż do uzyskania posortowanej tablicy.

Class Wstawianie
Jest tutaj zaimplementowany algorytm wstawianie przez proste wybieranie, jest on złożoności O(n^2) i porównuje się go często do układania kart w dłoni. Zaczynamy od drugiego elementu porównujemy go z pierwszym jeśli pierwszy jest mniejszy od prawego wtedy nie robimy nic, jeśli pierwszy jest większy od drugiego wtedy przesuwamy liczbę większą w prawo, do momentu gdy liczba trafi w odpowiednie miejsce, wykonujemy te porównania tak długo aż otrzymamy posortowany zbiór danych.

Class Wybieranie
Ostatni algorytmem jest sortowanie przez wybieranie również o złożoności O(n^2), polega ono na wyszukiwaniu najmniejszego elementu w tablicy i ustawianie go na początek, następnie czynność jest wykonywana ponownie (pomijając już elementy posortowane) tak długo, aż dostaniemy posortowane dane.

6. Testowanie
   Program został przetestowany na dużych, średnich i małych rozmiarach danych, wszystkie algorytmy sortują poprawnie z wyjątkiem Prostego Wstawiania czasem jedna liczba nie znajduje się w odpowiednim miejscu. dane posortowane zostają zapisane w odpowiednich plikach.

7. Wnioski

   Początkowo próbowałem zapisywać wyrazy w dwuwymiarowych tablicach char, niestety, dużo się z nimi męczyłem, żeby odpowiednio zaalokować dynamicznie pamięć, w wyniku czego pojawiały się częste błędy z wyciekami pamięci, zamiast nich użyłem tablicy stringów, które okazały się nieporównywalnie prostsze w obsłudze. Bardzo ciężki w implementacji okazał się algorytm qsort. Podczas tworzenia tego projektu nauczyłem się wiele na temat programowania obiektowego i zauważyłem jego zalety między innymi przejrzystość kodu, możliwość wykorzystywania metod klas w innych klasach za pomocą dziedziczenia co znacząco redukuje czas pracy nad projektem, bo programista nie musi ciągle przepisywać kodu tylko używa już istniejący. Z czasem nauki zauważyłem, że wiele mógłbym zmienić i poprawić w swoim projekcie, między innymi mógłbym zamiast wielokrotnego dziedziczenia zastosować polimorfizm, oraz użyć klasy generycznej, znacząco skróciło by to kod, a projekt działał by nadal tak samo.