**알고리즘 과제 4**

**2015003654 이호영**

1. 소스 코드 설명

main.c

파일 입출력을 담당하고 Sorting의 종류를 고를 수 있도록 한 소스 코드

mergeSort.c

merge 함수와 merge sort의 sequential version, recursion version을 구현한 코드

소스 코드는 Visual Studio Code 1.49.3 version에서 구현하였다.

Input\_i, input\_c: pdf 파일에 나온대로 입력파일을 만든 것

output\_i,output\_c: 입력파일에 대해서 merge sort를 진행한 것, 두 가지의 결과물이 같아 파일 하나만 첨부했다.

real\_data – 코로나바이러스 전세계 일일 확진자 변화량을 나타낸 숫자 데이터

real\_data2 – 2006년 가장 빈번하게 사용된 상위 10000개의 단어 데이터

output1, output2 그에 대한 정렬 결과

소스 참고는 Introduction to Algorithms 책과 수업 자료를 참고했습니다.

1. 가설 설정

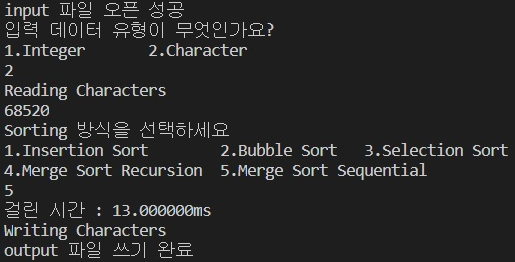
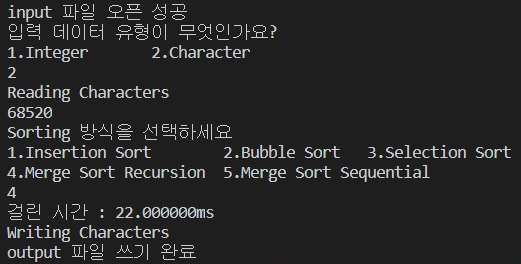
가설1: 데이터가 많아질수록 O(n^2)의 정렬 세 가지가 O(nlogn)인 Merge sort보다 훨씬 성능이 안 좋아 질 것이다.

근거: n^2의 기울기는 계속 가파르게 증가를 한다. 하지만 n \* logn에서 logn은 갈수록 기울기가 작아지기 때문에 data 양이 많아질수록 차이가 크게 벌어질 것이다.

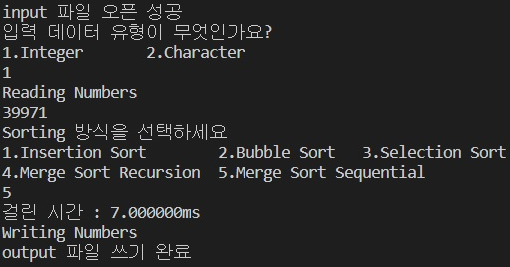
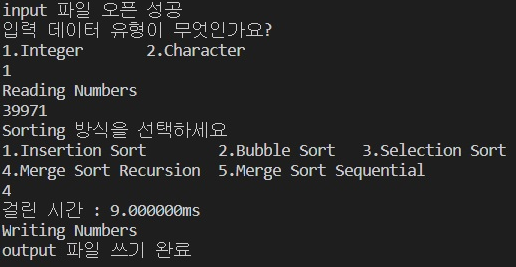
가설2: Merge Sort Recursion이 Sequential보다 성능이 더 좋게 나올 것 같다.

근거: Recursion의 코드의 길이도 짧고 Sequential에서는 반복문이 쓰여서 더 느릴 것 같다.

1. Data와 결과 및 가설 검증



W1에서 동일하게 썼던 data로 2006년에 가장 빈번하게 사용된 상위 만 개의 단어를 쓰인 빈도대로 내림차순으로 나열한 것을 배열한 것이다. Recursion 방식이 Sequential 방식에 비해 시간이 더 많이 소요된다.



이 data 역시 W1에서 동일하게 썼던 데이터로 2020년도 2월부터 7월까지 코로나 바이러스 확진자의 하루 증감량을 나타냈다. 문자 데이터보다 양이 적어서 그런지 두 개의 merge sort의 성능이 문자 정렬보다 차이가 덜 났다.

숫자와 문자 10개를 정렬하는 것의 결과는 데이터량이 적어서 0초에 수렴해 캡쳐를 하지 않았고 input과 output 결과만 첨부한다.

숫자 데이터(39971개)

Insertion Sort – 1014ms

Bubble Sort – 2958ms

Selection Sort – 2040ms

Merge Sort (Recursion) – 9ms

Merge Sort (Sequential) – 7ms

문자 데이터(68520개)

Insertion Sort – 3219ms

Bubble Sort – 14977ms

Selection Sort – 6077ms

Merge Sort (Recursion) – 22ms

Merge Sort (Sequential) – 13ms

가설 검증: 계산을 해보지 않아도 Merge Sort들의 소요시간은 2배 증가했지만 O(n^2) Sort들은 소요시간이 최소 3배에서 많게는 거의 5배까지 증가했다. 가설1은 맞는 것으로 결과가 나왔다.

Merge Sort의 Sequential version이 더 성능이 좋다고 나왔다. 거의 차이는 없지만 코드가 길다고 성능이 나쁜 것이 아니라는 것이다. 그 이유에 대해서 생각을 해보면 Recursion은 Stack에 함수를 계속 push하고 pop하기 때문에 조금 더 시간이 걸리지 않았을까 생각해본다. 그리고 Sequential의 반복문은 gap을 2씩 곱하는 연산이어서 logn번 연산을 할 것이고 반복문이 쓰였지만 수행 횟수는 그렇게 많지 않다. 그렇기에 가설2는 틀린 것으로 드러났다.