**Profiling Report**

**On my honour, I pledge that I have neither received nor provided improper assistance in the completion of this assignment.**

**서명: 김신후 학번: 21900136**

**이름: 김신후 학번: 21900136**

**목차**

**머리글: 아너코드와 출제자 정보**

1. **profiling.exe 의 출력 텍스트 파일 캡쳐본**
2. **5가지 케이스에 대한 성능 분석 표 a, b, estimate time 계산과정**
3. **5가지 케이스 비교 엑셀 차트와 그래프**
4. **5가지 케이스의 시간복잡도에 대한 출제자 의견**

**첨부파일**

**1. ProfilingReport.pdf**

**2. profiling\_result\_n40000.txt**

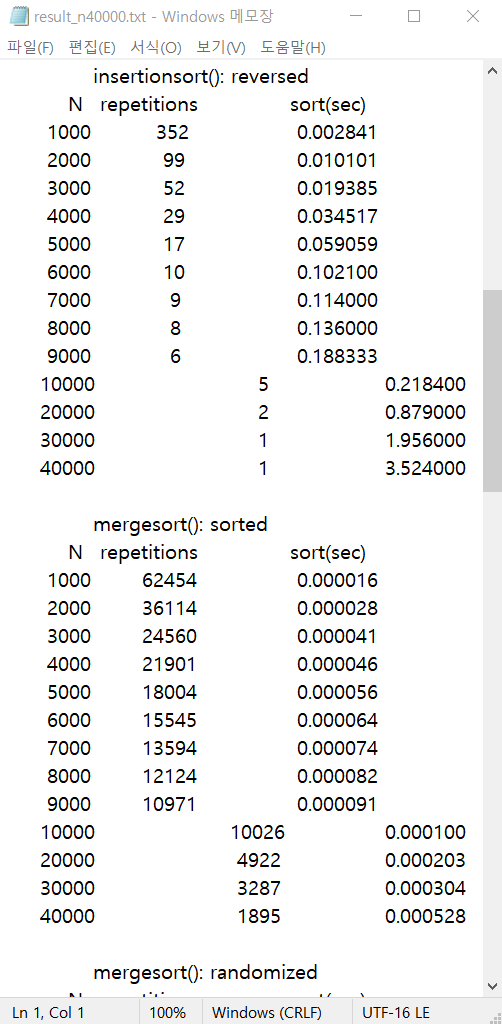
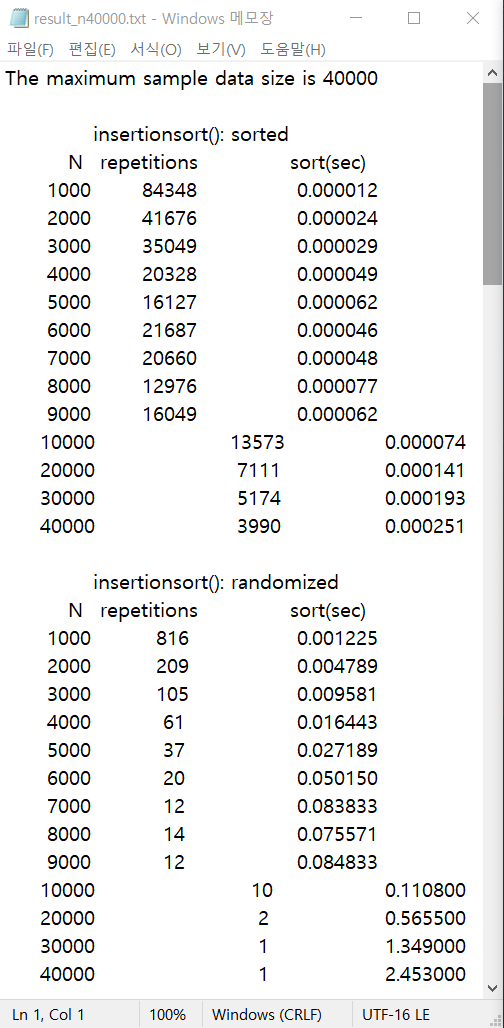
**3. 성장률 풀이과정.pdf**

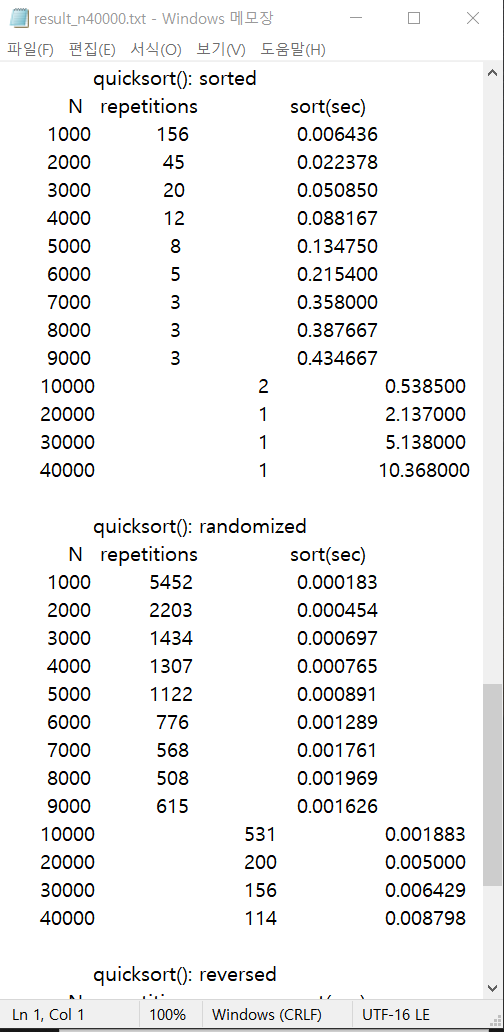
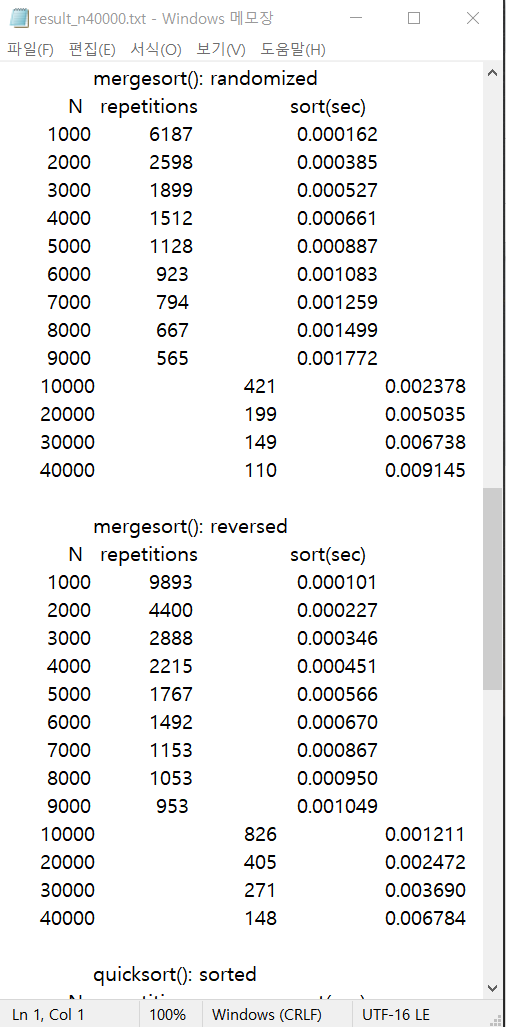
**4. 5가지 케이스 차트, 그래프, 성장률 계산.pdf**

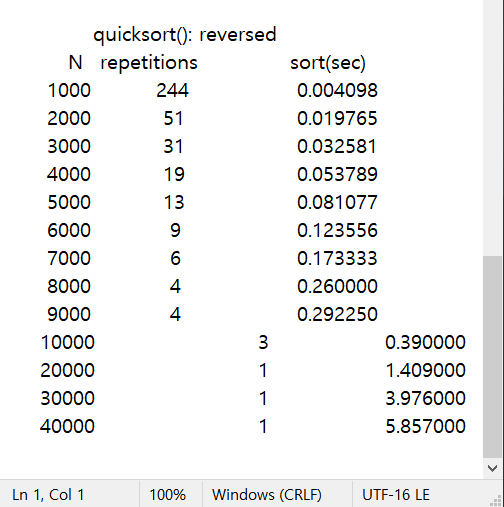
**5. FiveCaseN10^6.zip**

**profiling.exe 의 출력 텍스트 파일 캡쳐본**

아래 캡쳐본에서 차례대로 insertion, merge, quick sort 알고리즘이 sroted, randomized, reversed된 수열들을 정렬한 시간들을 확인할 수 있다.

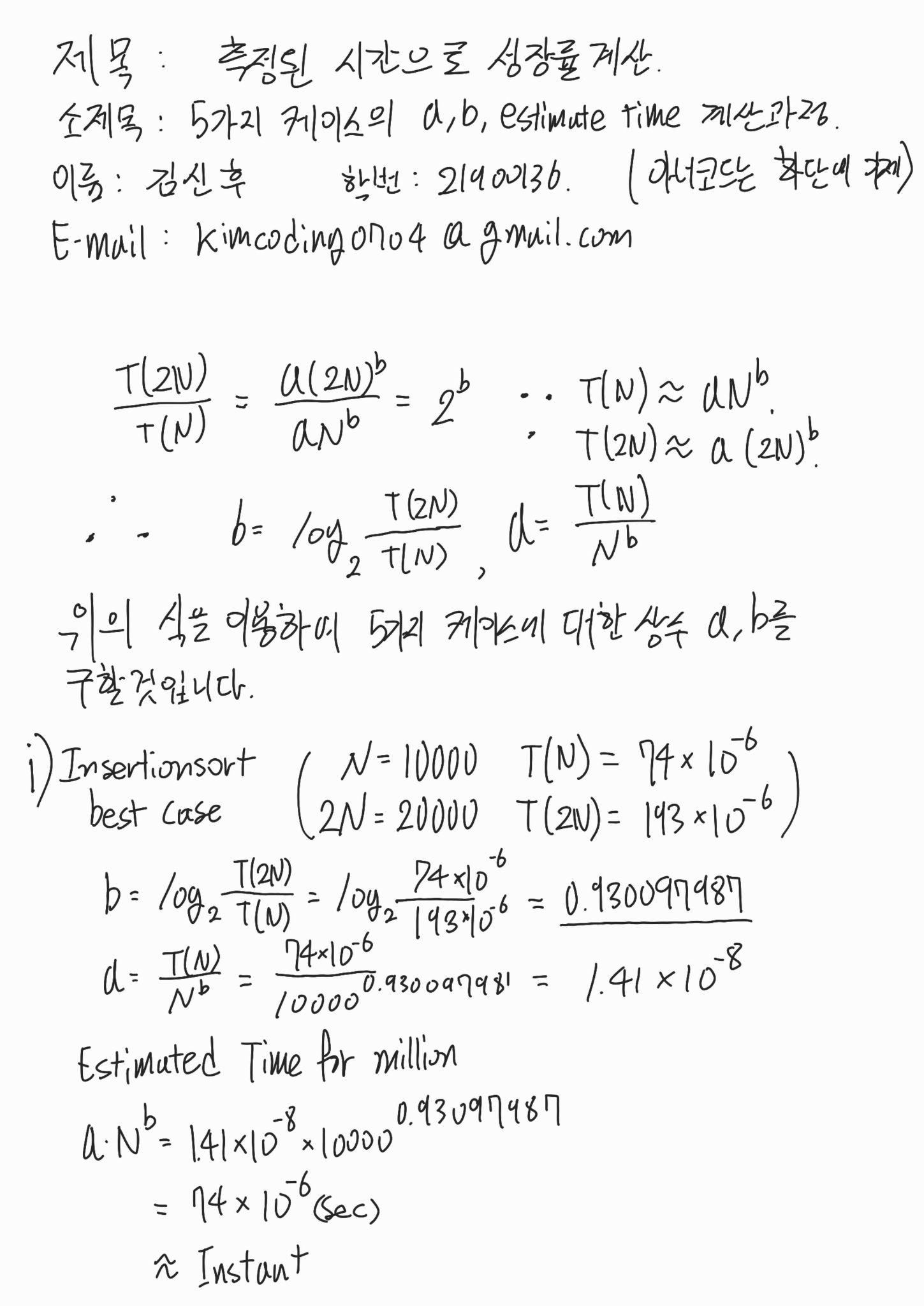


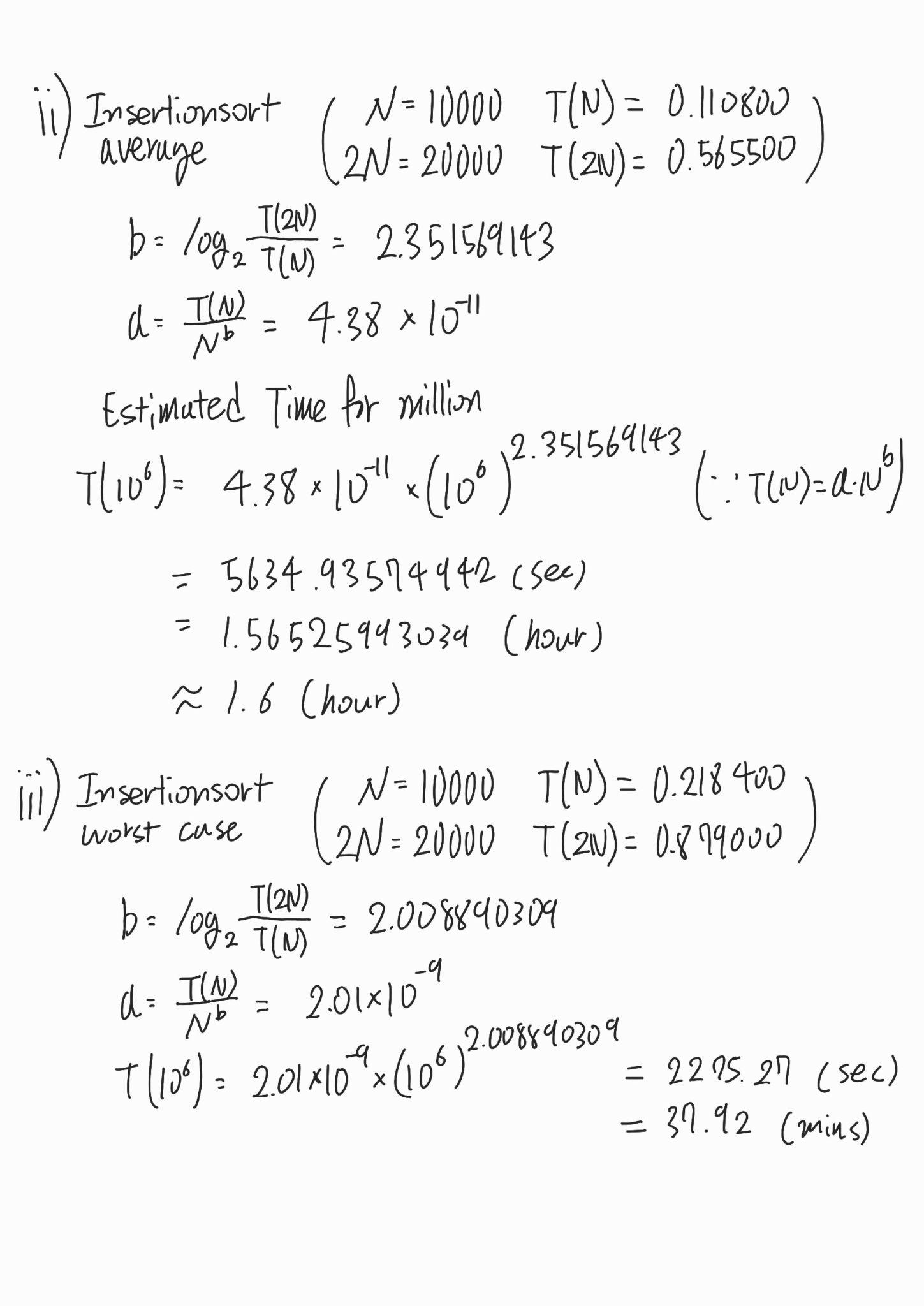


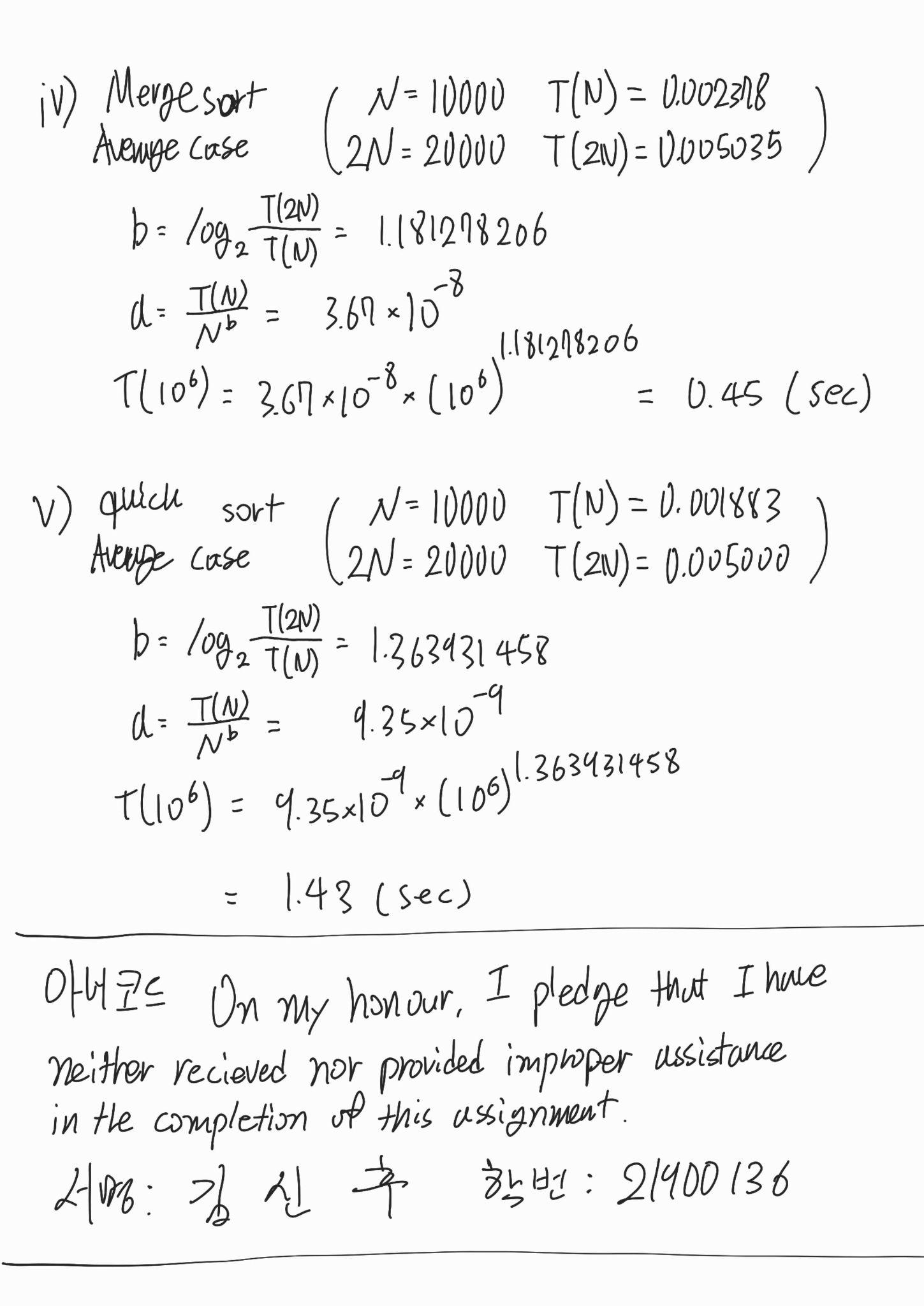


**5가지 케이스에 대한 성능 분석 표 a, b, estimate time 계산과정**

첨부파일 “성장률 풀이과정.pdf”





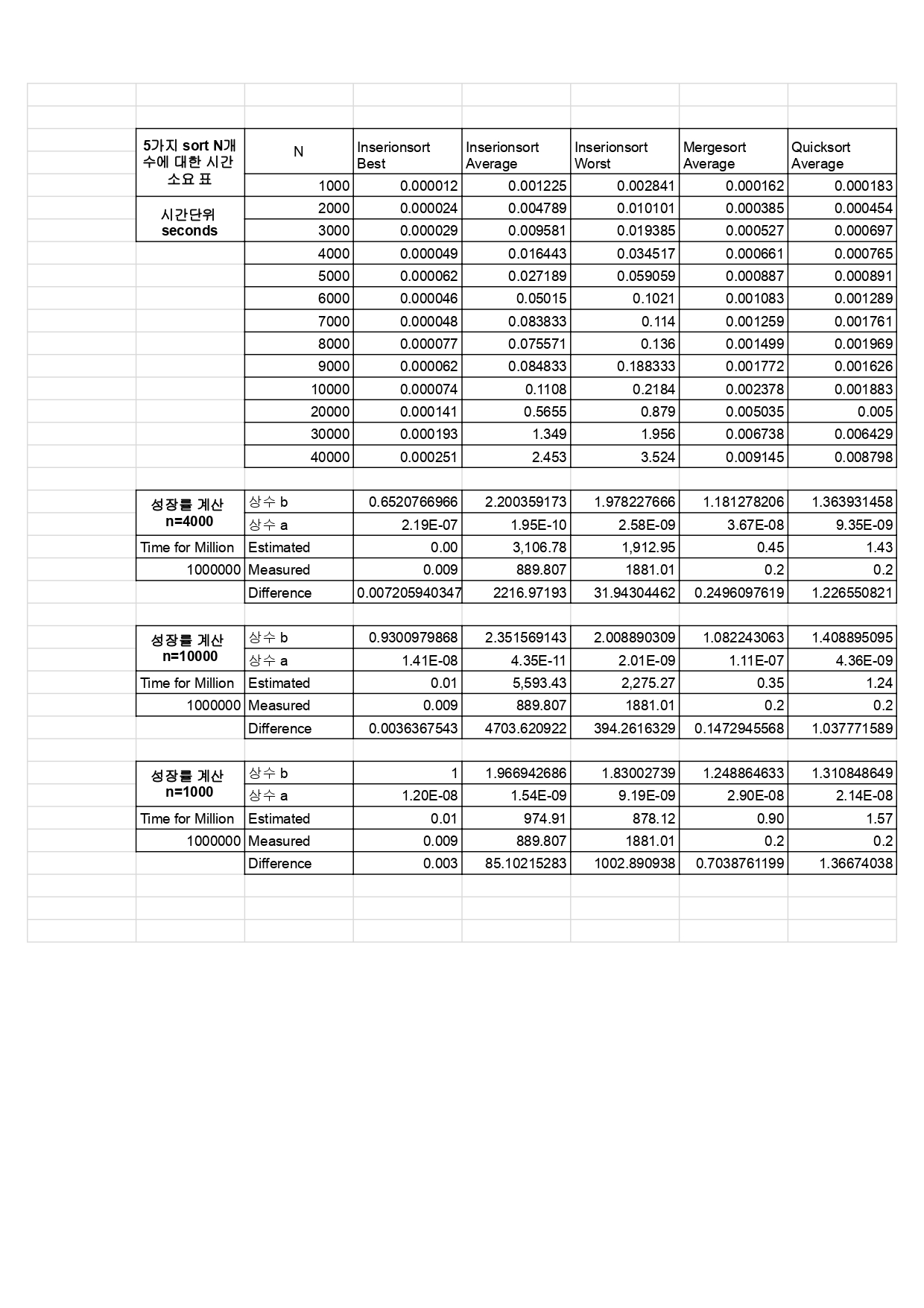


**5가지 케이스 비교 엑셀 차트와 그래프**

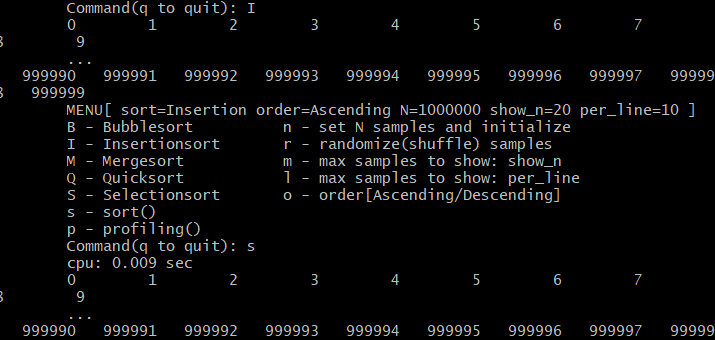
**driver.exe를 이용한 N이 100만일 때 계산 소요시간**

첨부파일

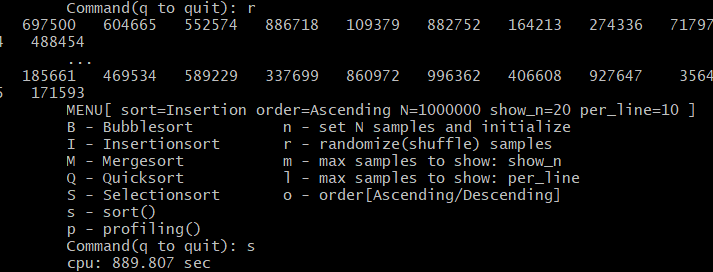
“5가지 케이스 차트, 그래프, 성장률 계산.pdf”, “driver.exe N=10^6 실행화면.zip”



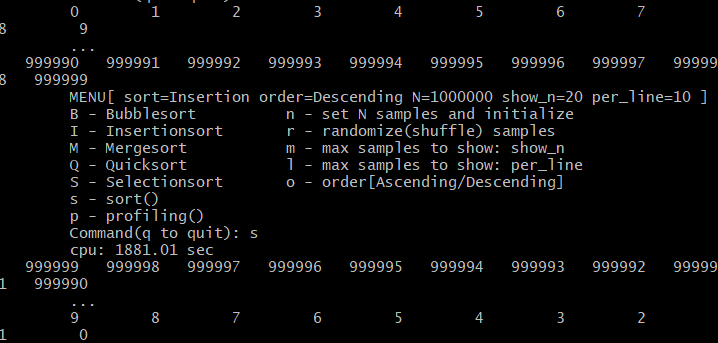
Insertion sort - best T(10^6) = 0.009 (sec)



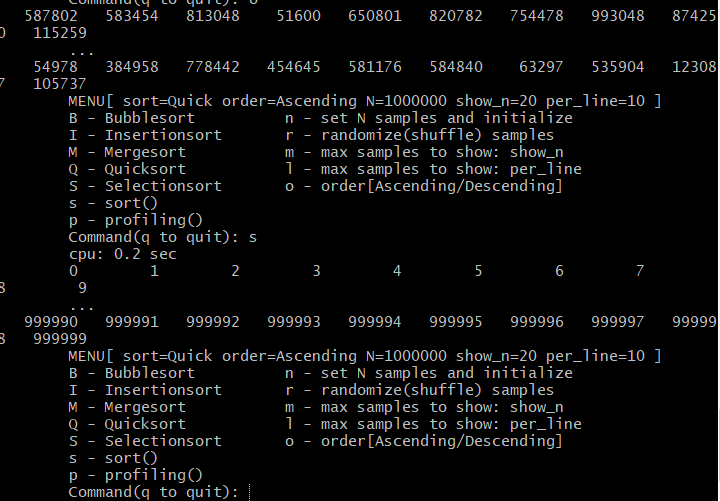
Insertion sort - average T(10^6) = 14.83 (min)



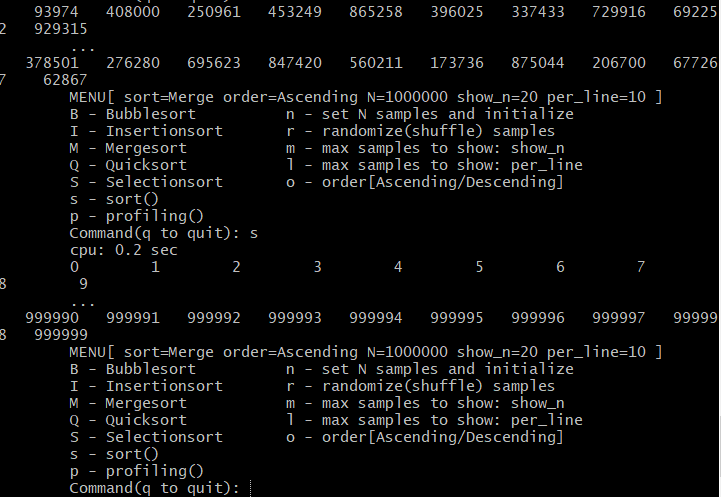
Insertion sort - worst T(10^6) = 31.35 (min)

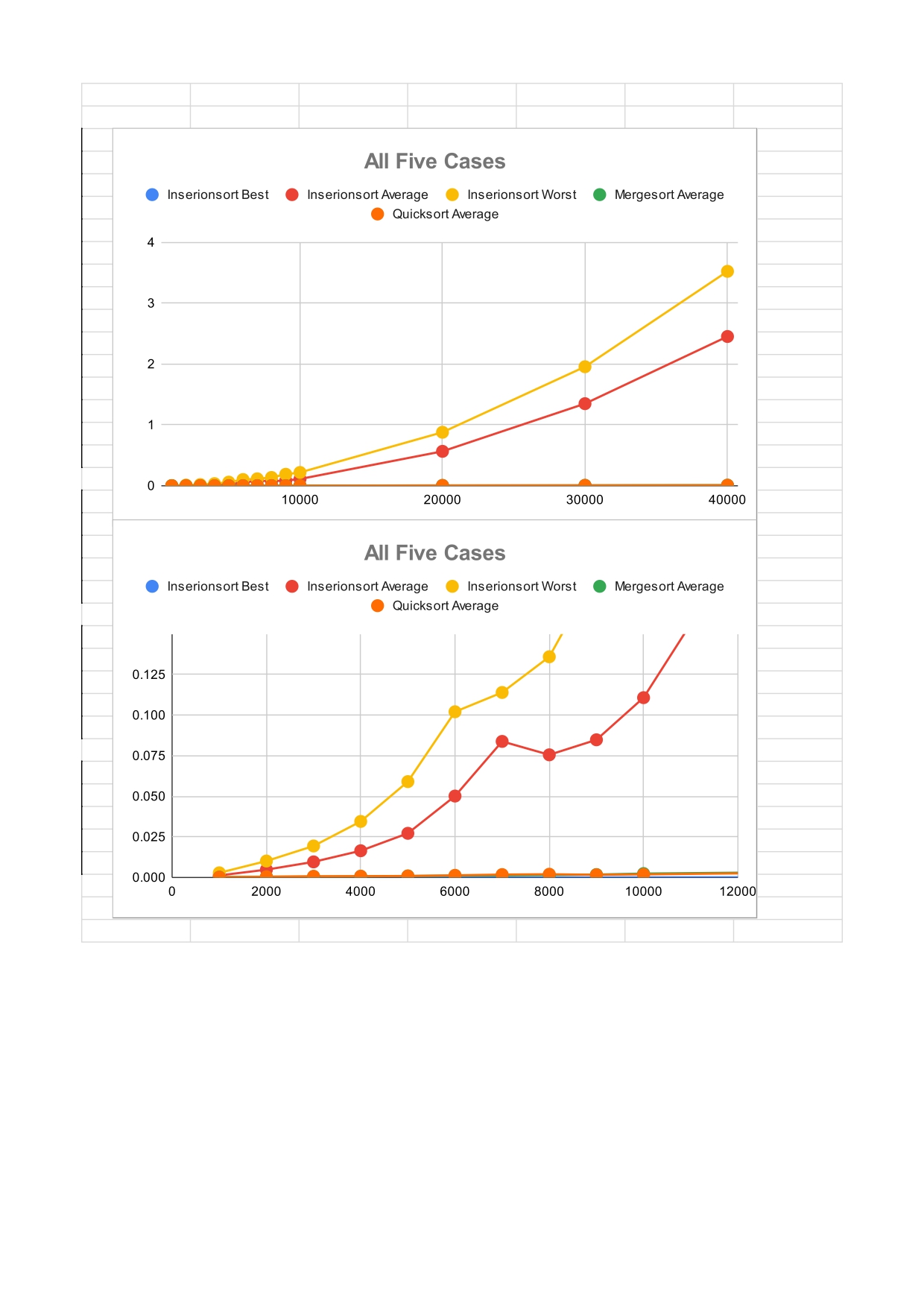


quick sort - average T(10^6) = 0.2 (sec)



merge sort - average T(10^6) = 0.2 (sec)





**5가지 케이스의 시간복잡도에 대한 출제자 의견**

이번 과제를 진행하면서 총 네 가지를 관찰하였다.

첫 번째로, 좋은 알고리즘은 무엇인가를 알아 보았다. 좋은 프로그램을 만들기 위해선 좋은 알고리즘이 필요하다. 좋은 알고리즘이 되기위한 요건 중 하나는 시간이 소모(time complexity)가 적은 것이다. 이번 과제를 진행하면서 세 가지 정렬 알고리즘을 time complexity 측면에서 비교해 보았다. 정렬할 숫자의 개수가 작을 때는 잘 느끼지못했지만, n이 커지면 커질 수록 selction 정렬 알고리즘의 계산 종료 속도가 급수적으로 길어졌음을 확인했다. 100만개의 어지럽혀진 숫자를 정렬하기 위해 quick과 merge 알고리즘은 0.2초가 걸린 반면에 selection 알고리즘은 약 14.8분이 걸렸다. selection 알고리즘은 n이 클때 좋은 프로그램을 위한 알고리즘은 아닐것이다.

두 번째로, 그렇다면 왜 selection 알고리즘이 quick과 merge 알고리즘보다 느릴 수 밖에 없는지 빅오 노테이션 O(N)을 통해서 알아보았다. selection 알고리즘은 O(N^2) time complexity를 가지고 있고 quick과 merge 알고리즘은 O(N log N) time complexity를 가지고 있다. 이것은 위의 엑셀 그래프를 보면 확인할 수 있는데, selection 알고리즘은 N^2 그래프를 보이고 quick과 merge 알고리즘은 N log N 그래프를 보인다. 그런데 한가지 의문점은 selection average 그래프에서 N이 8000일때가 N이 7000일때보다 수행속도가 빨랐다. N 8000 이후로는 N^2의 그래프를 다시 보인다. N이 7000에서 8000으로 넘어갈 때 컴퓨터가 계산 성능을 올렸을 것이다라고 추측하고 있다.

세 번째로, 수열의 정렬 방식이 정렬 알고리즘의 수행능력에 영향을 미친다는 것이다. selection 알고리즘을 예로들면 N이 100만일때, 세 종류의 수열 (정렬된 수열, 섞인 수열, 반대로 정렬된 수열), 에 대해서 정렬 소요 시간이 각각 0.009초, 14.8분, 31.3분으로 엄청난 차이를 보여주었다. 하지만 이와는 다르게 merge 알고리즘은 수열이 정렬이 되어있을 때가 정렬되어 있지 않을때보다 정렬하는 시간이 더 오래 소요된다. 따라서 절대적인 좋은 정렬 알고리즘을 찾는 것 보다 상황에 맞는 좋은 알고리즘을 선택하여 사용해야겠다. 예를 들면 아주 큰 N을 다룰 필요는 없고 사용할 수 있는 메모리가 여유롭지 않은 상황에서는 selection 알고리즘이 나을것이다.

마지막으로, T(N) ≈ a N^b 라는 식과 N이 비교적 작을 때 데이터를 활용해 T(N)에 대해서 일반식을 만들 수 있다는 것을 알아보았다. 이를 통해 N이 어떠할 때 시간이 얼마나 걸리는지 알아보기 위해 직접 측정하지 않아도 된다. 하지만, 어떠한 관측한 N을 이용해 상수를 구할 것인지, 컴퓨터 수행 능력은 일관적인지가 일반식에 영향을 미치는 것 같다. 상수 a, b를 구할 때, n=1000, n=4000, 그리고 n=10000 인경우로 총 세번 일반식을 구해 n=10^6일때 걸리는 시간을 예측해 보았는데, 각기 다른 정확도를 보인것을 확인할 수 있다.