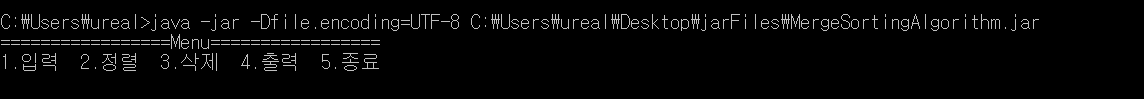
[ 멀티미디어 알고리즘 Report 2 ]

멀티미디어공학과

2019112541 양유진

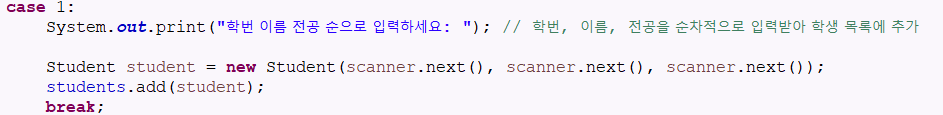
1. 프로그램 설명

Merge-Sorting 알고리즘과 Insertion-Sort 알고리즘을 각각 이용하여 텍스트 파일로 입력 받은 학생들의 학적부를 관리하는 프로그램이다. 원하는 학생의 정보(학번, 이름, 학과)를 입력하면 학생을 학적부에 추가할 수 있다. 학생의 학번을 입력하면 그 학생을 학적부에서 제거할 수 있고 학적부를 출력하면 현재 학적부의 상태를 확인할 수 있다.

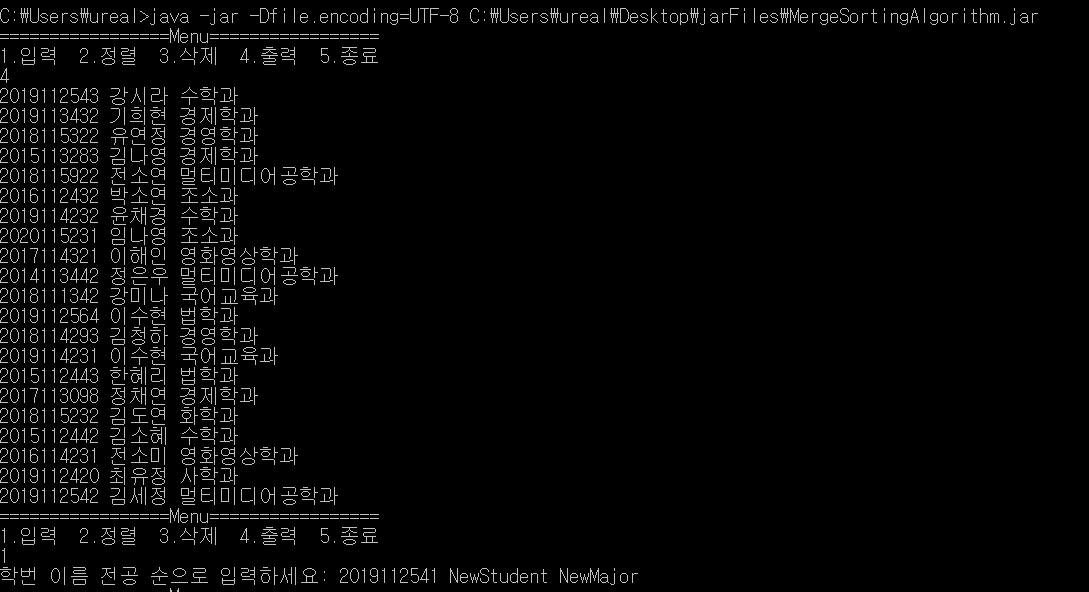
1. 실행

학적부 RegisterFile.txt에 학생이름과 전공이 한글로 되어있기 때문에 jar파일 실행 시 UTF-8로 encoding을 함께 해준다. txt파일 경로는 코드 내에서 설정해준다.

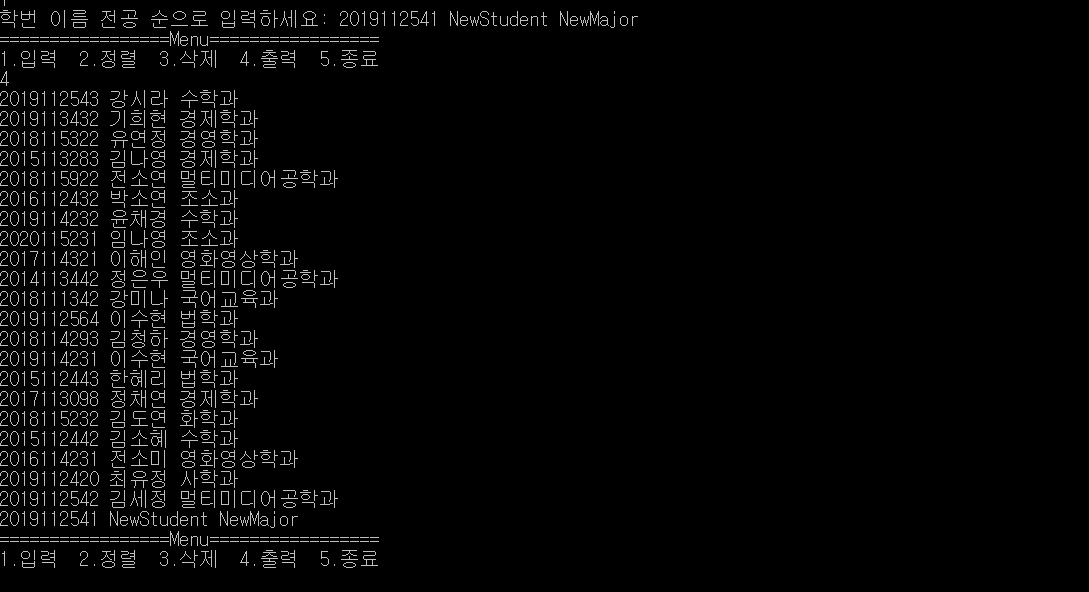
1. 입력



Student 객체를 입력 받아서 학생들의 명단인 students ArrayList에 추가해준다.

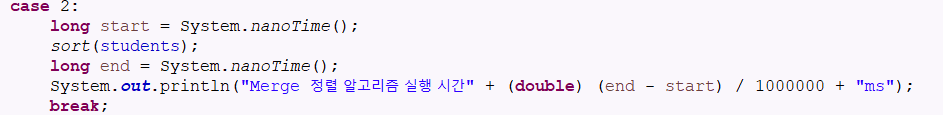


NewMajor의 전공을 가진 학번이 2019112541인 NewStudent라는 학생을 입력하였다.

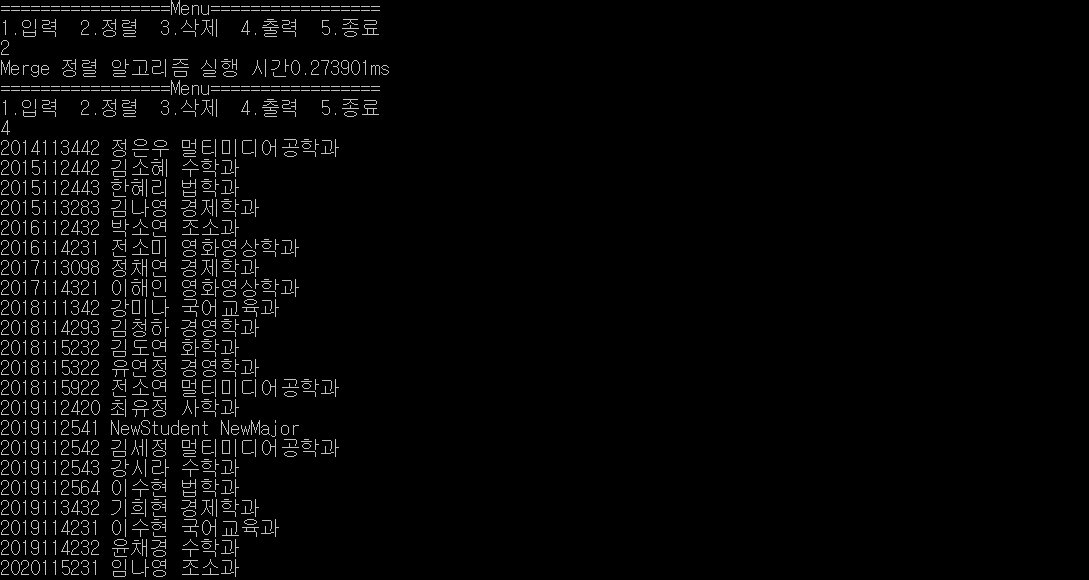


출력 시 가장 아래에 NewStudent라는 학생의 정보가 학적부에 추가된 것을 확인할 수 있다.

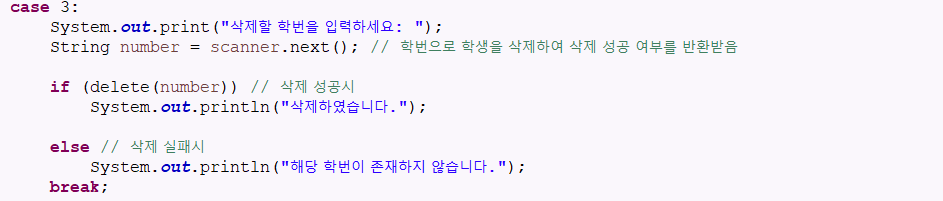
1. 정렬



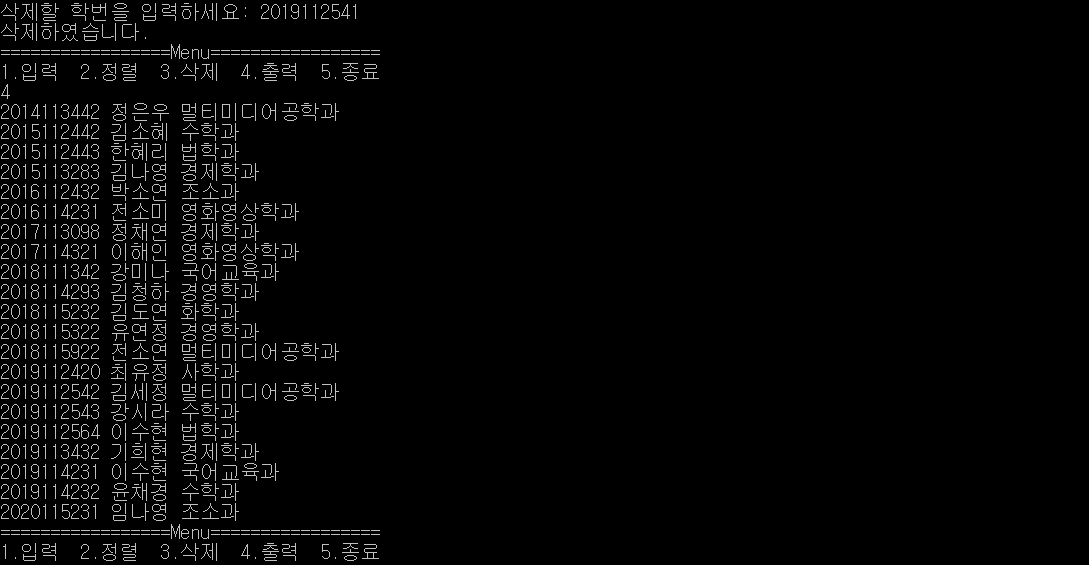
알고리즘 실행 시간 측정을 위해, 정렬 메소드 앞 뒤로, nanoTime()의 값을 저장하는 start, end 변수를 선언하였다. 측정 단위는 ‘ms’로 지정하였다.



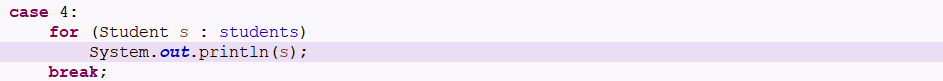
정렬 시 알고리즘 실행 시간이 출력된다. 예시는 Merge Sort를 사용하였지만 Insertion Sort도 같은 기능을 한다.

1. 삭제

학번을 입력 받아 학번이 list에 존재하는지 판단하고 존재하지 않을 시 삭제 실패로 간주하여 알람 문구가 뜬다.

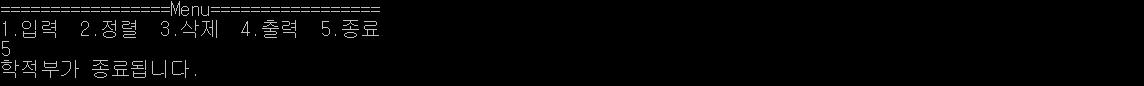


삭제할 학생의 학번을 입력하면 그 학생이 학적부에서 삭제되는 것을 확인할 수 있다.

1. 출력

students list에 있는 Student 객체들을 모두 출력한다.

1. 종료

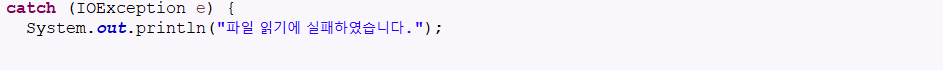


1. 예외처리

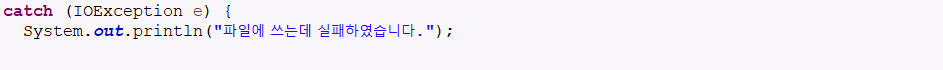
* menu에 없는 기능을 입력하였을 때



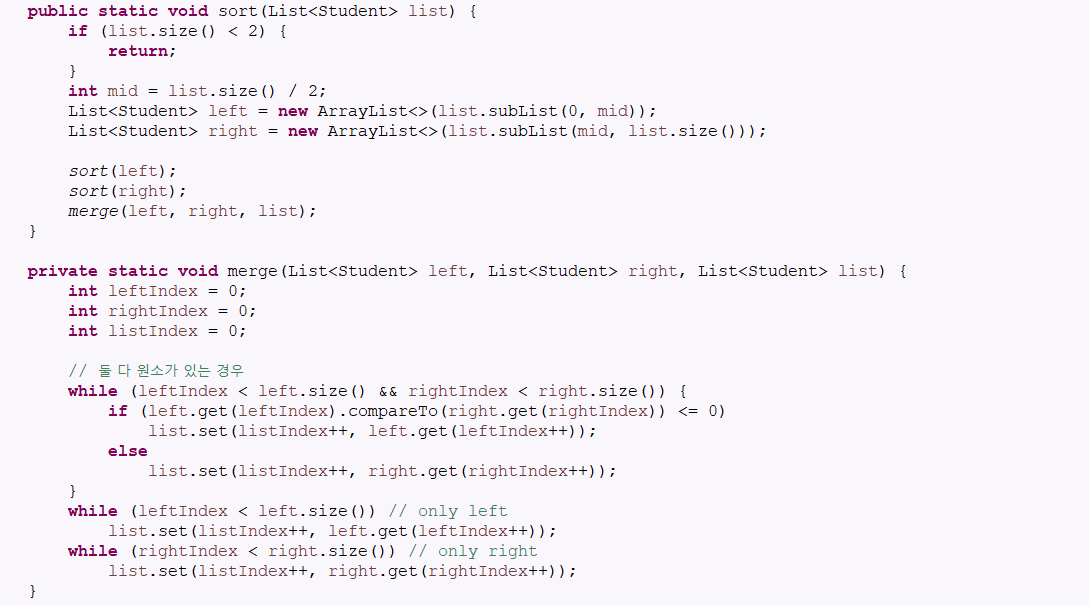
* text File이 존재하지 않거나 읽기를 실패할 때



* text File에 쓰기를 실패할 때

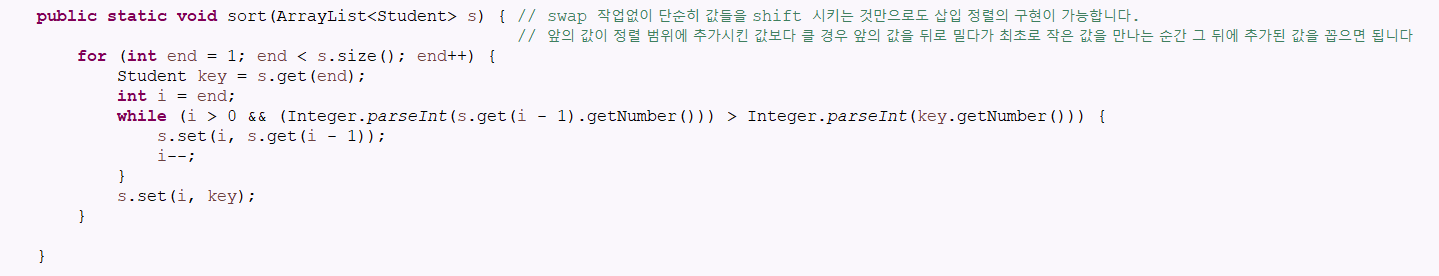


1. 알고리즘 설명
2. Merge-Sort Algorithm



Merge-Sort는 반으로 나누고 비교한 뒤, 나중에 합쳐서 정렬하는 정렬 알고리즘이다. Left List 와 Right List를 생성하여, Student list를 반으로 나눠주고 각각 비교하고 이 과정을 반복한다. 반복하고 나서, merge() 메소드를 이용하여 합쳐서 정렬해준다.

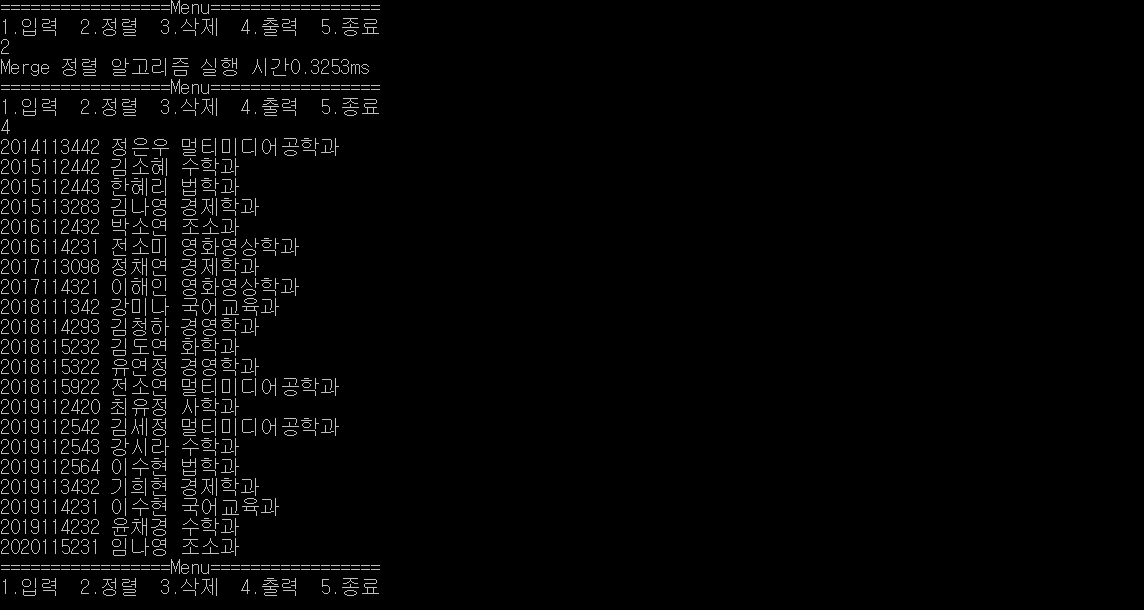
1. Insertion-Sort Algorithm



temp를 생성해서 swap 해주는 방법이 아닌, 최적화된 Insertion-Sort Algorithm을 구현하였다. 앞의 값이 정렬에서 추가시킨 값보다 클 경우, 값을 옮겨주면서 한 칸 씩 뒤로 이동시키다가 최초로 작은 값을 마주할 때 그 뒤 추가시킨 값을 선택한다.

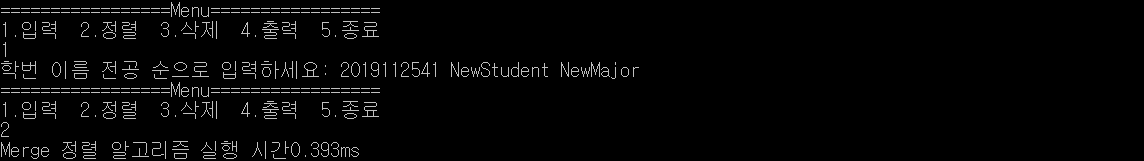
1. 실행 결과 및 분석

[1] Merge-Sort Algorithm

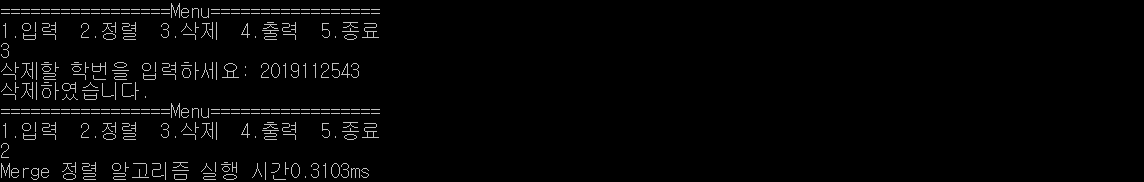
(1) 정렬 시 알고리즘 실행 시간 : 0.3253ms

1. 입력 후 정렬 시 알고리즘 실행 시간 : 0.4006ms

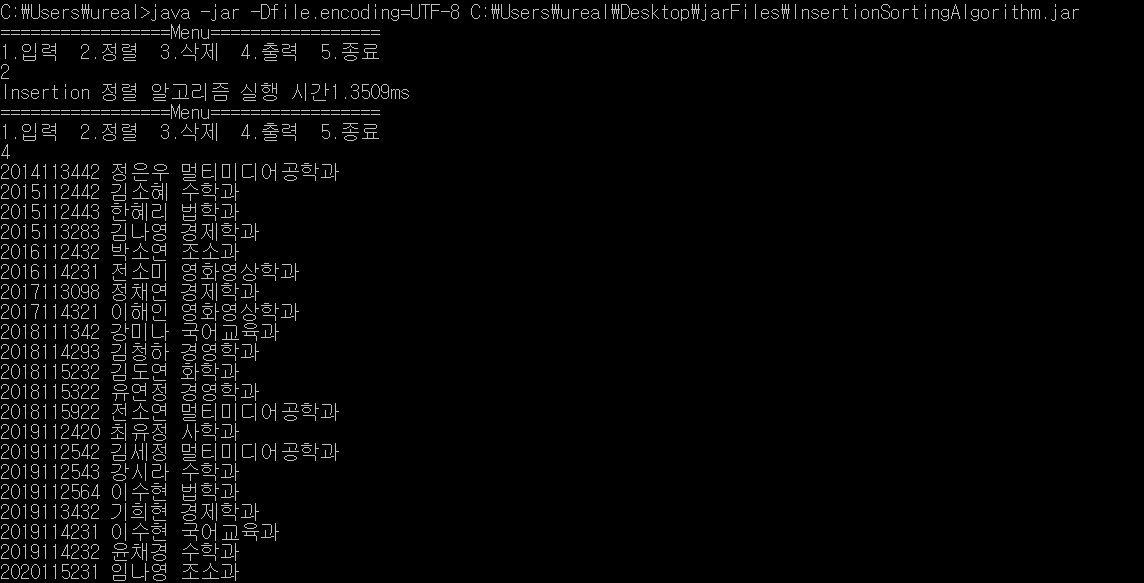
Student 객체가 추가되자 알고리즘 실행 시간이 길어졌다.



1. 삭제 후 정렬 시 알고리즘 실행 시간 : 0.3103ms

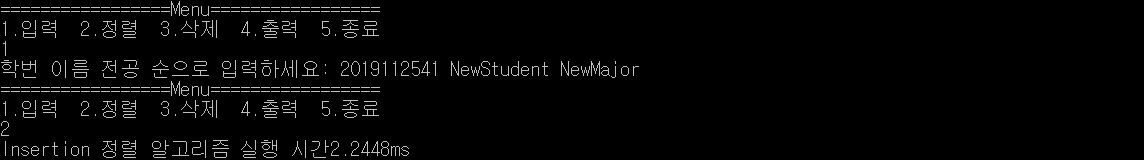
Student 객체를 하나 삭제하자 알고리즘 실행 시간이 짧아졌다.

[2] Insertion-Sort Algorithm

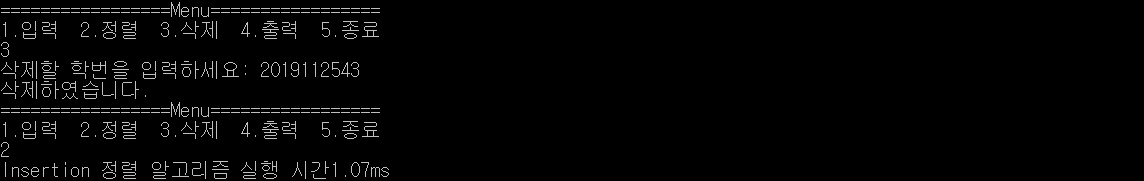
(1) 정렬 시 알고리즘 실행 시간 : 1.3509ms

1. 입력 후 정렬 시 알고리즘 실행 시간 : 2.2448ms

Student 객체가 추가되자 알고리즘 실행 시간이 길어졌다.



1. 삭제 후 정렬 시 알고리즘 실행 시간 : 1.07ms

Student 객체를 하나 삭제하자 알고리즘 실행 시간이 짧아졌다.

[3] 결론

실험 전, Merge Sort는 Ο(nlog₂n)의 시간 복잡도를 가진 상대적으로 빠른 알고리즘이고 Insertion Sort는 O(n²)의 시간 복잡도를 가진 상대적으로 느린 알고리즘이라고 알고 있었다. 그러므로 Merge Sort가 더 빠를 것이라고 예상하였지만 정렬을 시행할 Register.txt의 학생 명단은 22명으로, 입력이 비교적 적기 때문에 차이를 비교할 수 있을 지 의문이었다. 실험 결과, 적은 입력에도 불구하고 Merge Sort 알고리즘의 실행 시간과 Insertion Sort 알고리즘의 실행시간은 확실하게 차이가 난다는 것을 확인할 수 있었다. 또한 삭제 및 입력으로 입력 값을 조절하면 그에 따라서 실행 시간 역시 변화가 생긴다는 것을 확인 할 수 있었다.