1.테스트 환경

※조건

-한대의 pc를 사용하여 pc의 성능에 의한 속도차이가 없도록 한다.
-pc의 휴식기를 설정하여 속도저하의 영향을 최소화하여 정렬을 실행한다.

※PC환경

-OS : window 10 Home
-컴파일러 : JDK javac
-CPU : Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80GHz
-RAM : 8.00GB

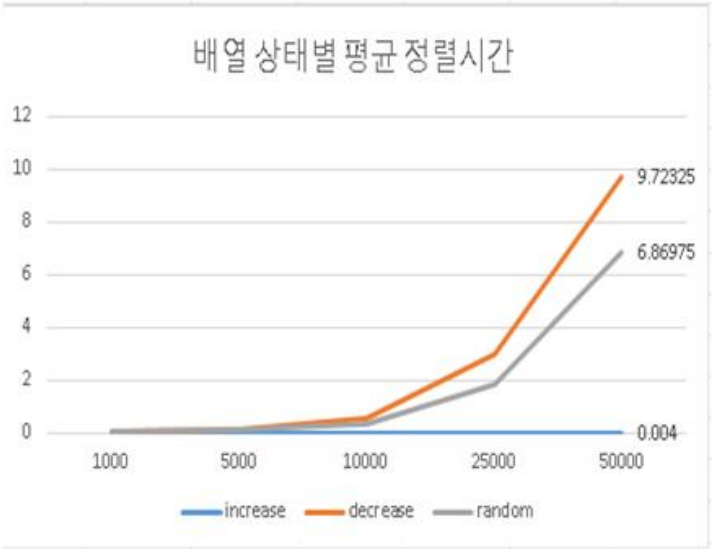
※성능 비교방법

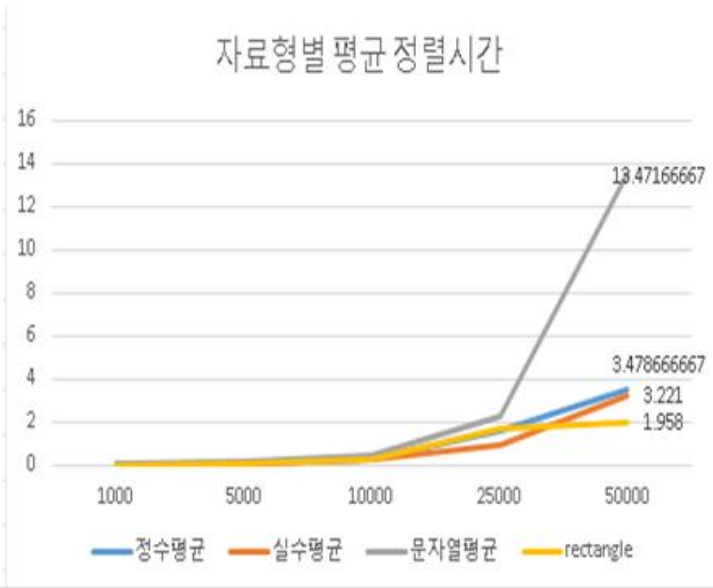
-정렬 성능 측정에 사용되는 데이터는 정수, 실수, 문자열, 클래스(Rectangle) 데이터 타입을 쓴다.
-위 데이터 타입을 오름차순, 내림차순, 랜덤상태로 구성된 배열을 대상으로 한다.
-위 데이터를 담는 배열은 각 1천, 5천, 1만, 2만5천, 5만의 크기를 가진다.

3.정렬분석

※Insertion Sort

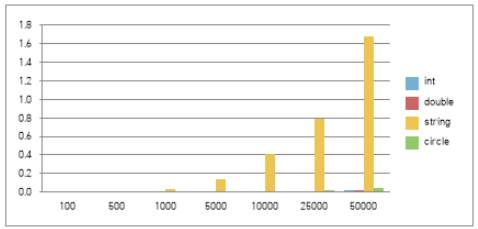
-시간측정 그래프





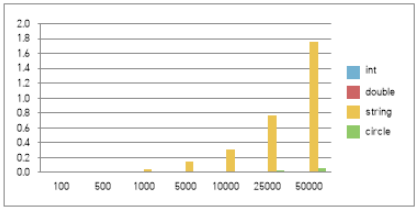
3-10) Heap Sort

- random Data



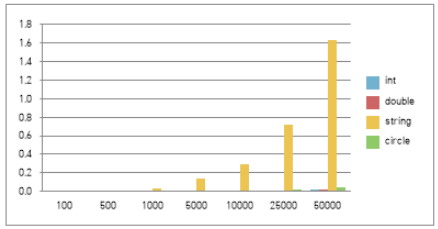
	int	double	string	circle
100	0.000	0.000	0.002	0.000
500	0.000	0.000	0.005	0.000
1000	0.000	0.001	0.030	0.001
5000	0.001	0.001	0.135	0.003
10000	0.003	0.004	0.407	0.008
25000	0.007	0.007	0.784	0.021
50000	0.015	0.015	1.667	0.045

-Increase Data



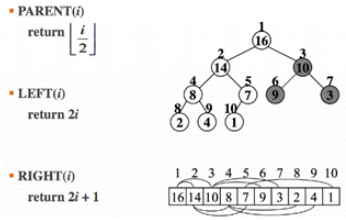
	int	double	string	circle
100	0.000	0.000	0.002	0.000
500	0.000	0.000	0.003	0.000
1000	0.000	0.000	0.029	0.002
5000	0.001	0.001	0.141	0.004
10000	0.003	0.003	0.298	0.008
25000	0.005	0.006	0.762	0.02
50000	0.012	0.011	1.744	0.044

-Decrease Data



	int	double	string	circle
100	0.000	0.000	0.001	0.000
500	0.000	0.000	0.005	0.000
1000	0.000	0.000	0.028	0.001
5000	0.001	0.001	0.139	0.003
10000	0.003	0.002	0.289	0.007
25000	0.005	0.006	0.714	0.019
50000	0.013	0.015	1.621	0.041

분석 :
Heap Sort는 최대 힙 트리나 최소 힙 트리를 구성해 정렬을 하는 정렬 방법이다. 내림차순 정렬을 위해서는 최대 힙을 구성하고 오름차순 정렬을 위해서는 최소 힙을 구성하여 사용한다.



- heap 구조를 사용하기 위한 배열 구조

배열을 Max Heap 구조로 변환한다. O(logN) - ①
Root Node를 마지막 Node와 교체한 후, 해당 노드는 정렬된 값으로 간주하고 나머지 배열에 대해서만 ①의 과정을 다시 진행. O(N) - ②
전체 자료가 정렬될 때까지 ① 과 ②의 과정을 반복한다.
따라서 O(NlogN)의 시간 복잡도를 가지게 된다.

Heap Sort의 경우 Random, Increase, decrease 세 정렬의 속도도 모두 비슷하게 측정 되었다. 즉, 시간복잡도가 모두 O(NlogN) 으로 동일하다.

따라서 Heap Sort는 최악의 경우도 O(NlogN)으로 보장 되어있기 때문에, $O(N^2)$이 되는 상황을 만들지 않고 평균적으로 빠르게 정렬할 때 상당히 용이 할 것이다.