**4. Sorting**

1. 목표

여러가지 정렬 알고리즘을 구현하고 성능을 비교하면서 알고리즘의 중요성을 깨닫도록 합니다.

1. 문제

아래와 같은 여러 가지 정렬 알고리즘을 구현하고 비교해 봅니다.  
아래의 소팅 중에 배우지 않은 것들은 다른 곳에서 소팅하는 방법을 공부해보고 작성해 봅시다.

* + [B] bubble sort
  + [I] insertion sort
  + [S] shell sort (Textbook Ch.10 - Programming Problems 8)
  + [H] heap sort (Textbook Ch. 11, binary heap)
  + [M] merge sort
  + [Q] quick sort
  + [R] radix sort

1. **뼈대 코드(Skeleton Code) :**[**여기**](https://soar.snu.ac.kr/course/ds/20102/assignments/4/SortingTest.txt) **[중요] 이번 숙제는 뼈대 코드를 필히 사용해야 합니다. (같은 입출력 여건에서의 객관적인 비교를 위함)  
   뼈대 코드를 사용하지 않을 경우 입출력 문제나, 잘못된 시간체크 연산에 의해 불이익을 받을 수 있습니다.  
   또한 뼈대 코드의 내용을 이해해서 자신의 것으로 만들기 바랍니다.  
   뼈대 코드의 마지막 부분에 각각의 소팅에 대한 함수가 있습니다. 거기서부터 구현을 시작하면 됩니다.**

1. 입력 형식 : 숫자 입력 -> 소팅 방법

1. 입력은 숫자의 나열로 이루어지며, 첫 줄에 총 숫자의 갯수, 둘째줄부터 각각의 숫자가 입력됩니다.

예) 9              (총 숫자의 갯수)  
    100  
    7  
    925  
    -234  
    10  
    -9999  
    12738  
    239  
    -2391          (여기까지 총 9개)  
  
  
2. 숫자의 나열 대신 난수를 입력할 수도 있습니다. 이 때는 첫 줄에 r (갯수) (최소값) (최대값) 이 오고 끝납니다.

예) r 1000 -30000 30000        (-30000 에서 30000 까지(-30000과 30000도 포함) 1000개의 숫자를 랜덤하게 생성하고 그것을 입력으로 대신한다)

3. 숫자의 나열이 정의 되었으면 다음 줄에서 소팅 방법이 입력됩니다.  
    방법은 B, I, S, H, M, Q, R (2번 항목의 소팅 종류 참조) 중의 한 글자로 입력됩니다.

4. 소팅 방법이 입력되면 소팅을 수행하고 수행결과를 출력합니다.  
    출력한 다음 다시 소팅 방법을 입력받습니다. (3번으로 돌아감)  
    만약 소팅 방법에 X가 입력되었다면 종료합니다.

**\* 모든 숫자는 정수이며 int의 범위 내에 속합니다. 숫자들의 총 갯수는 1000000개 이하입니다.**

1. 출력 형식 : 2가지 입력 방법(숫자의 나열 or 난수)에 따라 다르게 출력

1. 숫자의 나열이 입력되었다면 정렬된 숫자들을 오름차순(작은값에서 큰값으로) 으로 한 값당 한줄에 출력합니다.(정렬 결과 출력)  
   (작성한 코드가 정렬을 잘 수행하는지를 확인하고자 하는 목적)

예) -9999  
     -2391  
     -234  
     7  
     9  
     10  
     100  
     239  
     925

2. 난수가 입력되었다면 정렬을 수행하되 결과는 출력할 필요 없고 대신 수행 시간(ms)를 출력합니다.  
   (작성한 코드가 각각의 정렬 방법에 따라 어느 정도의 시간이 걸리는지를 알아보고자 하는 목적)  
   뼈대 코드에 구현이 되어 있습니다.

1. 입출력 예제  
     
       $ java SortingTest                        <- 프로그램 실행  
       5                                                 <- 이렇게 입력(숫자가 총 5개라는 뜻)  
       3                                                 <- 이렇게 입력(첫번째 숫자)  
       1                                                 <- 이렇게 입력(두번째 숫자)  
       9                                                 <- 이렇게 입력(세번째 숫자)  
       3                                                 <- 이렇게 입력(네번째 숫자)  
       4                                                 <- 이렇게 입력(다섯번째 숫자)  
       Q                                                 <- 이렇게 입력(Quick Sort를 수행하라)  
       1                                                 <- 이렇게 출력(QuickSort로 오름차순 정렬한 첫번째 숫자)  
       3                                                 <- 이렇게 출력(QuickSort로 오름차순 정렬한 두번째 숫자)  
       3                                                 <- 이렇게 출력(QuickSort로 오름차순 정렬한 세번째 숫자)  
       4                                                 <- 이렇게 출력(QuickSort로 오름차순 정렬한 네번째 숫자)  
       9                                                 <- 이렇게 출력(QuickSort로 오름차순 정렬한 다섯번째 숫자)  
       B                                                 <- 이렇게 입력(Bubble Sort를 수행하라)  
       1                                                 <- 이렇게 출력(BubbleSort로 오름차순 정렬한 첫번째 숫자)  
       3                                                 <- 이렇게 출력(BubbleSort로 오름차순 정렬한 두번째 숫자)  
       3                                                 <- 이렇게 출력(BubbleSort로 오름차순 정렬한 세번째 숫자)  
       4                                                 <- 이렇게 출력(BubbleSort로 오름차순 정렬한 네번째 숫자)  
       9                                                 <- 이렇게 출력(BubbleSort로 오름차순 정렬한 다섯번째 숫자)  
       X                                                 <- 이렇게 입력(이제 종료해라)  
       $                                                 <- 프로그램 종료  
     
       $ java SortingTest                        <- 프로그램 실행  
       r 100 -200 200                               <- 이렇게 입력(숫자가 총 100개이며 범위는 -200 에서 200 사이에서 랜덤하게 생성)  
       Q                                                 <- 이렇게 입력(Quick Sort를 수행하라)  
       100 ms                                         <- 이렇게 출력(QuickSort를 수행하는데 걸린 시간.)  
       B                                                 <- 이렇게 입력(Bubble Sort를 수행하라)  
       10000 ms                                      <- 이렇게 출력(BubbleSort를 수행하는데 걸린 시간.)  
       X                                                 <- 이렇게 입력(이제 종료해라)  
       $                                                 <- 프로그램 종료