프로그래밍 과제 02

1. harry_full.txt 파일에 등장하는 모든 단어들을 다음의 6가지 알고리즘을 이용해 사전식 순서로 정렬하는데 걸리는 시간을 초 단위로 측정하여 출력하는 프로그램을 작성하라. 파일은 총 125553개의 단어로 이루어져 있다.

   (1) 버블 정렬(bubble sort)

   (2) 삽입 정렬(insertion sort)

   (3) 합병정렬(merge sort)

   (4) 빠른정렬(quicksort) – 마지막 값을 피봇으로 선택

   (5) 힙정렬(heap sort)

   (6) C, C++ 혹은 Java등 자신이 사용하는 언어의 표준 라이브러리가 제공하는 정렬 알고리즘

   입출력 시간 등을 제외하고 순수하게 정렬 알고리즘이 실행되는 시간만을 측정해야 한다.

2. 주소록 파일이 있다. 파일의 한 라인은 한 사람에 대한 이름, 회사(company), 주소(address), 우편번호(zip code), 전화번호, 이메일 주소의 6가지 필드로 구성된다. 필드와 필드는 하나의 '|' 문자로 구분되어 있다. 샘플파일을 다운로드하여 사용하라. 이 주소록 파일을 읽은 후 각각의 사람을 하나의 객체로 저장한 후 다음과 같이 사용자의 명령을 수행하는 프로그램을 작성하라. 단, 반드시 사용하는 언어의 표준 라이브러리가 제공하는 정렬 기능을 사용하라. 정렬 알고리즘을 직접 구현해서는 안된다. 불필요한 공백이나 탭(tab) 문자 등이 저장되지 않도록 주의하라.

```
$ read address.txt
$ sort -name      // 사람들을 name의 알파벳 순으로 정렬한다.
$ print           // 정렬된 순서대로 모든 사람에 대한 모든 정보를 다음과 같이 화면에 출력한다.
Abel
    Company: Rangoni Of Florence
    Address: 37275 St  Rt 17m M Middle Island Suffolk NY
    Zipcode: 11953
    Phones: 631-335-3414
    Email: amaclead@gmail.com
Adelina
    Company: Courtyard By Marriott
    Address: 80 Pittsford Victor Rd #9 Cleveland Cuyahoga OH
    Zipcode: 44103
    Phones: 216-230-4892
    Email: adelina_nabours@gmail.com
…

$ sort -address   // 사람들을 주소의 알파벳 순으로 정렬한다.
$ print           // 정렬된 순서대로 모든 사람에 대한 모든 정보를 위와 동일한 형식으로 출력한다.
…

$ exit
```

3. 최대우선순위 큐를 구현하는 단순한 방법의 하나는 정렬되지 않은 배열을 사용하는 것이다. 새로운 원소는 항상 배열의 맨 끝에 삽입하고, 최대값 삭제는 최대값을 $O(N)$ 시간에 찾아서 삭제하고 배열의 마지막 원소를 최대값이 있던 위치로 이동하여 배열의 중간에 빈 칸이 없도록 처리한다. 이렇게 단순하게 배열로 구현된 경우와 힙(heap)을 이용하여 우선순위큐를 구현 한 경우의 실행속도를 비교하라. 성능의 비교는 다음과 같이 한다. 우선 N개의 0에서 N사이의 정수를 랜덤하게 생성하여 우선순위큐에 삽입한다. 그런 다음 M번의 삽입 혹은 최대값 삭제 연산을 연속하여 실행하는데 걸리는 총 시간을 측정한다. M번의 연산 각각은 우선 1/2의 확률로 삽입 연산인지 혹은 최대값 삭제 연산인지를 결정한 후, 삽입 연산인 경우 다시 0에서 N사이의 정수를 랜덤하게 생성하여 삽입할 데이터를 결정한다. M과 N이 각각 100,000인 경우에 대해서 두 방법의 실행시간을 측정하여 출력하라. 최대한 공정한 비교가 되도록 하라.

```
void test(int N, int M) {
    for (int i=0; i<N; i++)
        pqueue.add(rd.nextInt(N));   // pqueue is the priority queue to test

    for (int i=0; i<M; i++) {
        if (rd.nextInt(2)==0 || pqueue.empty() )     // add
            pqueue.add(rd.nextInt(N));
        else
            pqueue.extractMax();
    }
}
```

4. [분할정복법] 추가 강의 슬라이드에서 설명한 2차원 평면에서 n개의 점들이 입력으로 주어질 때 볼록 헐(convex hull)을 구하여 출력하는 알고리즘을 구현하라. 시간복잡도는 $O(n \log n)$이 되어야 한다. 입력은 `input.txt` 파일로 부터 받는다. 파일의 첫 줄에는 입력 점의 개수 $n \leq 100$이 주어지고 이어진 n줄에 한 줄에 하나씩 점의 좌표가 주어진다. 출력은 볼록헐에 포함된 점들의 좌표를 시계방향 혹은 반시계방향 순서로 출력한다. 시작점이 누가 되는지는 상관없다.

입력 예(INPUT.TXT)	출력
13 643 516 575 567 623 556 579 474 683 451 703 537 662 582 628 625 753 630 740 493 695 610 562 652 601 733	753 630 740 493 683 451 579 474 562 652 601 733

입력 예(INPUT.TXT)	출력
12 115 110 162 141 210 129 103 243 149 199 136 172 200 185 173 244 50 151 180 109 97 194 125 55	173 244 200 185 210 129 125 55 50 151 103 243
14 148 139 32 183 85 128 138 82 287 99 227 141 102 165 153 206 200 105 290 203 202 177 216 40 81 262 83 66	290 203 287 99 216 40 83 66 32 183 81 262
20 242 195 76 60 132 141 329 94 237 136 199 82 139 220 398 266 316 166 192 165 190 129 264 94 293 210 126 87 247 294 348 219 258 165 192 219 246 247 327 254	247 294 398 266 329 94 76 60 139 220