# 자료구조응용

## 19. Sorting: insertion sort, quick sort

1. 다음과 같이 사용자로부터 데이터를 입력받아서 삽입정렬(insertion sort)을 수행한 결과를 출력하라. 각 레코드는 (key, name, grade)의 필드로 구성된다. 키값에 대하여 올림차순으로 삽입정렬을 하시오.

#### (1) 입력파일(input.txt)

```
7
10 송중기 95
35 조인성 89
25 김수미 59
50 홍길동 33
15 아이유 65
11 박용우 78
33 장윤정 67
```

#### (2) 실행순서

- ① 입력파일로부터 데이터를 읽어 들여 구조체 배열에 저장한다.
- ② key에 대해 내림차순으로 삽입정렬을 실행한다.
- ③ 정렬된 순서대로 (key, name, grade)를 화면에 출력한다.
- ④ 정렬결과를 파일(output.txt)에 저장한다.

```
void insert(element e, element a[], int i)
{/* insert e into the ordered list a[1:i] such that the
    resulting list a[1:i+1] is also ordered, the array a
    must have space allocated for at least i+2 elements */
    a[0] = e;
    while (e.key < a[i].key)
    {
        a[i+1] = a[i];
        i--;
    }
    a[i+1] = e;
}</pre>
```

Program 7.4: Insertion into a sorted list

```
void insertionSort(element a[], int n)
{/* sort a[1:n] into nondecreasing order */
   int j;
   for (j = 2; j <= n; j++) {
      element temp = a[j];
      insert(temp, a, j-1);
   }
}</pre>
```

Program 7.5: Insertion sort

### (3) 실행 예

2 입력리스트의 데이터를 파일로 입력받아 쉘정렬 수행결과를 단계적으로 실행의 예와 같이 출력 하시오.

#### (1) 입력파일(input.txt)

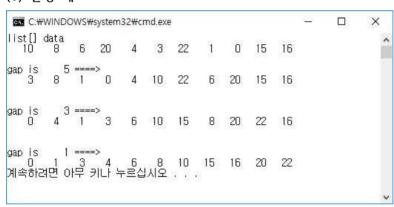
 11

 10 8 6 20 4 3 22 1 0 15 16
 ※ 첫 줄은 레코드의 정렬할 키의 개수

#### (2) 실행순서

- ① 입력파일(input.txt)로부터 데이터를 읽어 들여 배열 list에 저당한다.
- ② 각 원소에 대해 쉘 정렬을 실행한다.
- ③ gap의 크기에 따라 정렬된 결과를 출력한다.

#### (3) 실행 예



- 3. 입력리스트의 데이터를 파일로 입력받아 퀵정렬 수행결과 및 quickSort 함수호출 회수를 구하여 출력하라. 단, 각 레코드는 하나의 int형 key 필드로 구성되어 있다.
- (1) 입력파일(input.txt)

```
      11

      12 2 16 30 8 28 4 10 20 6 18
      ** 첫 줄은 레코드의 정렬할 키의 개수
```

#### (2) 실행순서

- ① 입력파일(input.txt)로부터 데이터를 읽어 들여 구조체 배열 a에 저장한다.
- ② 각 레코드의 key에 대해 퀵정렬을 실행한다.
- ③ 정렬된 key값 및 quickSort 함수호출 회수를 화면에 출력하라.
- ④ 정렬결과를 파일(output.txt)에 저장한다.

```
void quickSort(element a[], int left, int right)
[/* sort a[left:right] into nondecreasing order
   on the key field; a[left].key is arbitrarily
   chosen as the pivot key; it is assumed that
   a[left].key <= a[right+1].key */
  int pivot, i, j;
  element temp;
  if (left < right) {
     i = left; j = right + 1;
    pivot = a[left].key;
     do {/* search for keys from the left and right
            sublists, swapping out-of-order elements until
            the left and right boundaries cross or meet */
        do i++; while (a[i].key < pivot);
        do j--; while (a[j].key > pivot);
        if (i < j) SWAP(a[i],a[j],temp);</pre>
     } while (i < j);
     SWAP(a[left],a[j],temp);
     quickSort(a,left,j-1);
     quickSort(a,j+1,right);
}
```

Program 7.6: Quick sort

#### (3) 실행 예



#### ■ 제출 형식

- 공학인증 시스템(ABEEK)에 과제를 올릴 때 제목:

- 1차 제출: 학번 이름 DS-19(1), 2차 제출: 학번 이름 DS-19(2)

- 솔루션 이름 : DS-19 - 프로젝트 이름 : 1, 2, 3

- 실행화면을 캡쳐하여 한글파일에 추가 후 솔루션 폴더에 포함.

- 한글 파일명 : 학번\_이름\_실습결과.hwp

- 솔루션 폴더를 압축하여 게시판에 제출할 것.

- 압축 파일 명: 학번\_이름\_DS-19.zip

- 제출은 2회걸쳐 가능(수정 시간 기준으로 처리)

1차 종료시 까지 제출 (100%) 2차 종료시 까지제출 (80%)