자료구조응용
08. 연결 리스트 : 기초, 스택, 큐

1. 다음과 같이 정렬되지 않는 점수(정수 데이터)를 입력하면서 정렬된 Linked List를 만들고 실행예와 같이 수행되는 프로그램을 작성하라.

(1) 실행 순서

① 입력파일("input.txt")로 부터 데이터를 입력받으면서 **정렬된 Linked List**를 만든다. 입력 데이터는 정렬되지 않은 값으로 중복 가능하다.

input.txt
50 80 30 20 19 90
30 55 77 30 87 7

② Linked List의 처음부터 끝까지 노드의 데이터를 출력한다.
출력 형태 : (노드주소, 데이터 필드, 링크값)

③ 성적이 홀수인 노드를 Linked List에서 삭제한다.

④ Linked List의 처음부터 끝까지 노드의 데이터를 출력한다.
출력 형태 : (노드주소, 데이터 필드, 링크값)

⑤ Linked List를 모두 삭제한다.

(2) 구현 세부사항

① 구조체 정의문은 다음과 같다.

```
typedef struct listNode *listPointer;
typedef struct listNode {
        int data;
        listPointer link;
        } listNode;
listPointer first = NULL;
```

② 함수

- find : insert 위치를 찾는 함수를 새롭게 정의
- insert, printList : Program 4.2, 4.4를 참고하여 **수정**
- delete : Program 참고

```
void insert(listPointer *first, listPointer x)
{/* insert a new node with data = 50 into the chain
    first after node x */
   listPointer temp;
   MALLOC(temp, sizeof(*temp));
   temp→data = 50;
   if (*first) {
      temp→link = x→link;
      x→link = temp;
   }
   else {
      temp→link = NULL;
      *first = temp;
   }
}
```

Program 4.2: Simple insert into list

```
void insert(...)
{
  // empty list에 대해
  ① 첫 노드로 추가

  // non-empty list에 대해
  ② 첫 노드로 추가
  ③ 첫 노드 아닌 경우
}
```

```
void delete(listPointer *first, listPointer trail,
                                listPointer x)
{/* delete x from the list, trail is the preceding node
    and *first is the front of the list */
   if (trail)
     trail→link = x→link;
   else
     *first = (*first)→link;
   free(x);
}
```

Program 4.3: Deletion from a list

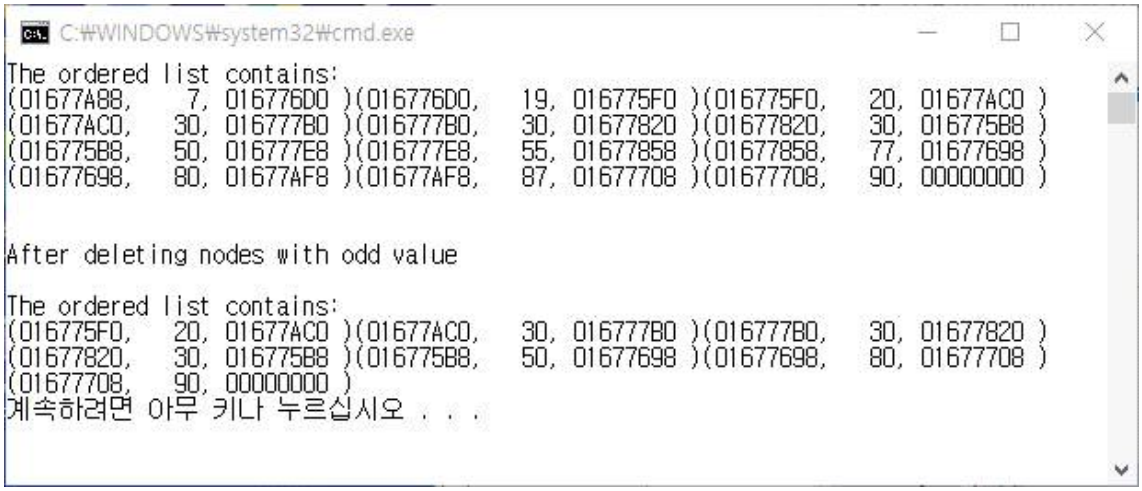
```
void printList(listPointer first)
{
   printf("The list contains: ");
   for (; first; first = first→link)
      printf("%4d",first→data);
   printf("\n");
}
```

Program 4.4: Printing a list

③ 데이터 입력 및 정렬된 리스트 만들기

```
fscanf_s(...);
while( !feof(fp) )
{
        find(....)          // find insert position x
        insert(...);        // insert data first after node x.
        fscanf_s(...);
}
```

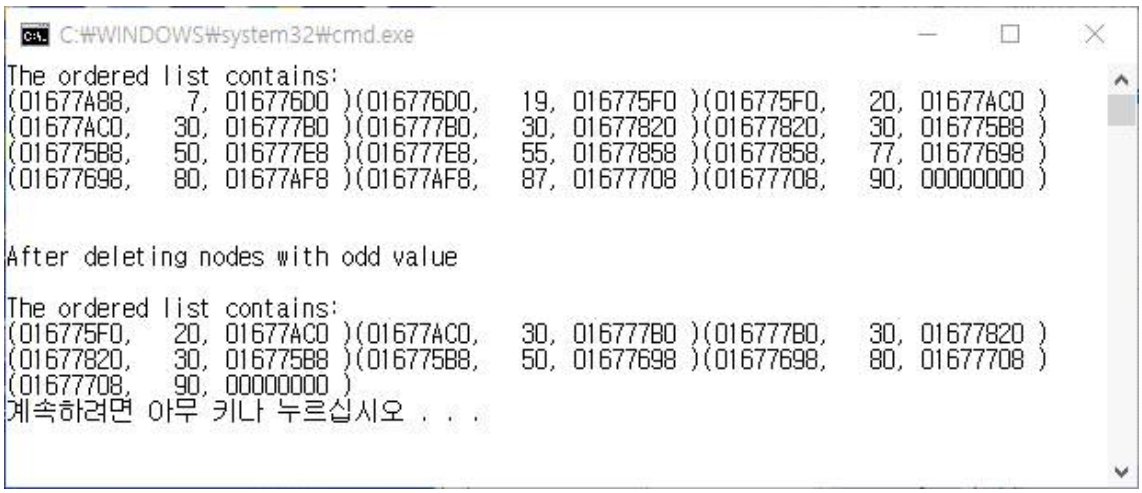
실행 예



2. 위 1번 문제를 구현한 소스를 다음과 같은 구조체 정의를 사용하는 프로그램으로 수정하라. 실행결과는 1번과 완전히 동일해야 한다. (listPointer 라는 자료형을 정의하지 않고 프로그램, listNode 라는 자료형을 사용하여 구현)

```
typedef struct listNode {
        int data;
        struct listNode *link;
        } listNode;
listNode *first = NULL; // or struct listNode *first = NULL;
```

실행 예



3. [Linked Stacks] 다음과 같은 스택을 생성하고 실행하는 프로그램을 작성하라. 이를 위해, push, pop, stackEmpty 함수를 구현하여야 한다.

(1) 실행 순서

① 학번 순으로 미리 정렬된 (과목번호, 학번, 성적)의 쌍으로 구성된 입력파일("input.txt")을 입력받으면서 입력되는 순서대로 각 과목 별로 Linked Stack에 저장 하시오. (과목이 3개이므로 3개의 스택이 필요함)

```
0 1 95
1 1 80
2 1 89
0 2 45
1 2 81
0 3 45
1 3 12
2 3 33
0 4 99
1 4 94
2 4 91
0 5 67
2 5 49
```

② 각 과목 별로 학번의 역순으로 노드의 데이터(학번, 성적)를 출력하라.

(2) 구현 세부사항

```
#define MAX_STACKS 3
typedef struct {
        int id;          //학번
        int grade;       //성적
        } element;
typedef struct stack *stackPointer;
typedef struct stack {
        element data;
        stackPointer link;
        }Node;
stackPointer top[MAX_STACKS];
/*************************************************************
top[i] = NULL, 0 ≤ i < MAX_STACKS // initial condition
top[i] = NULL, iff the ith stack is empty // boundary condition
*************************************************************/
```

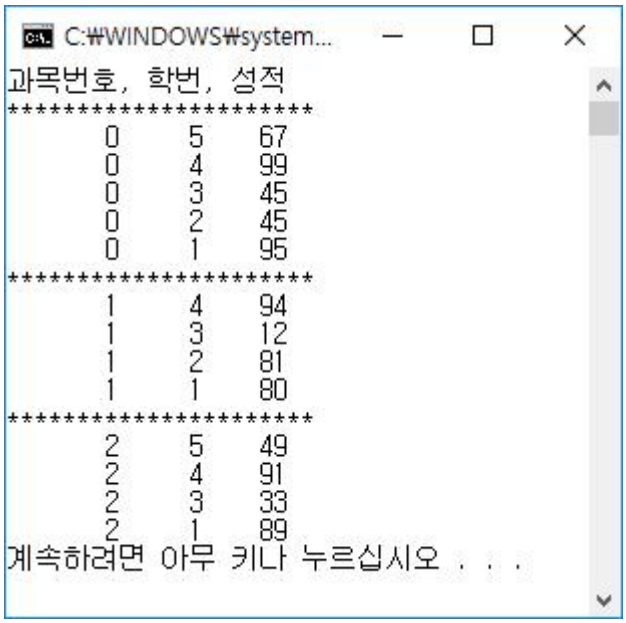
```
void push(int i, element item)
{/* add item to the ith stack */
   stackPointer temp;
   MALLOC(temp, sizeof(*temp));
   temp→data = item;
   temp→link = top[i];
   top[i] = temp;
}
```

Program 4.5: Add to a linked stack

```
element pop(int i)
{/* remove top element from the ith stack */
   stackPointer temp = top[i];
   element item;
   if (!temp)
     return stackEmpty();
   item = temp→data;
   top[i] = temp→link;
   free(temp);
   return item;
}
```

Program 4.6: Delete from a linked stack

실행 예



4. [Linked Queues] 다음과 같은 큐를 생성하고 실행하는 프로그램을 작성하라. 이를 위해, addq, deleteq, qEmpty 함수를 구현하여야 한다.(3점)

(1) 실행 순서

① 입력파일(“input.txt”)로 부터 학번 순으로 미리 정렬된 데이터를 입력받으면서 순서대로 Linked Queue를 구현한다. (과목번호, 학번, 성적)의 쌍으로 데이터들이 입력되며 각 과목별로 큐에 저장된다.

```
0 1 95
1 1 80
2 1 89
0 2 45
1 2 81
0 3 45
1 3 12
2 3 33
0 4 99
1 4 94
2 4 91
0 5 67
2 5 49
```

② 각 과목 별로 학번 순으로 노드의 데이터(학번, 성적)를 출력하라.

(2) 구현 세부사항

```
#define MAX_QUEUES 3
typedef struct {
        int id;             //학번
        int grade;          //성적
        } element;
typedef struct queue *queuePointer;
typedef struct queue {
        element data;
        queuePointer link;
        }Node;
queuePointer front[MAX_QUEUES], rear[MAX_QUEUES];
/*************************************************************
front[i] = NULL, 0 ≤ i < MAX_QUEUES // initial condition
front[i] = NULL, iff the ith queue is empty // boundary condition
*************************************************************/
```

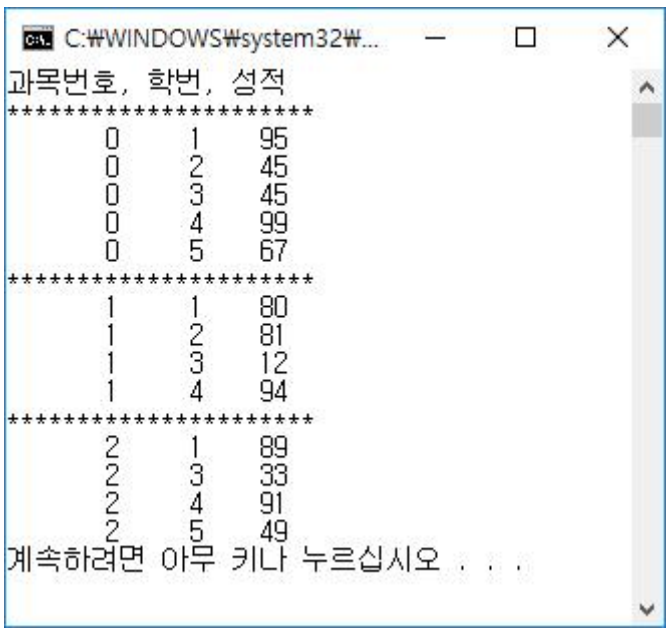
```
void addq(int i, element item)
{/* add item to the rear of queue i */
   queuePointer temp;
   MALLOC(temp, sizeof(*temp));
   temp→data = item;
   temp→link = NULL;
   if (front[i])
      rear[i]→link = temp;
   else
      front[i] = temp;
   rear[i] = temp;
}
```

Program 4.7: Add to the rear of a linked queue

```
element deleteq(int i)
{/* delete an element from queue i */
   queuePointer temp = front[i];
   element item;
   if (!temp)
      return queueEmpty();
   item = temp→data;
   front[i]= temp→link;
   free(temp);
   return item;
}
```

Program 4.8: Delete from the front of a linked queue

실행 예



■ 제출 형식

- 공학인증 시스템(ABEEK)에 과제를 올릴 때 제목:
- 1차 제출: 학번_이름_DS_08(1), 2차 제출: 학번_이름_DS_08(2)
- 솔루션 이름 : DS_08
- 프로젝트 이름 : 1, 2, 3, 4
- 실행화면을 캡쳐하여 한글파일에 추가 후 솔루션 폴더에 포함.
- 한글 파일명 : 학번_이름.hwp
- 솔루션 폴더를 압축하여 제출할 것.
- 솔루션 압축 파일 명:

  1차 제출: 학번_이름_DS_08(1).zip, 2차 제출: 학번_이름_DS_08(2).zip

- 제출은 2회 걸쳐 가능(수정 시간 기준으로 처리)