

# 자료구조응용

## 05. 스택과 큐

1. [정적할당배열의 스택] 다음과 같은 스택을 생성하고 실행 예와 같이 수행되는 프로그램을 작성하라. 이를 위해, push, pop, stackEmpty, stackFull, sprint(스택의 내용을 출력) 함수를 구현하여야 한다.

[자료형과 함수의 정의]

```
#define MAX_STACK_SIZE 5
typedef struct {
    int id;                // unique id
    char name[MAX_NAME_SIZE]; // last name
} element;
element stack[MAX_STACK_SIZE];
int top = -1;
```

```
void push(element item)
{
    /* add an item to the global stack */
    if (top >= MAX_STACK_SIZE-1)
        stackFull();
    stack[++top] = item;
}
```

**Program 3.1:** Add an item to a stack

```
element pop()
{
    /* delete and return the top element from the stack */
    if (top == -1)
        return stackEmpty(); /* returns an error key */
    return stack[top--];
}
```

**Program 3.2:** Delete from a stack

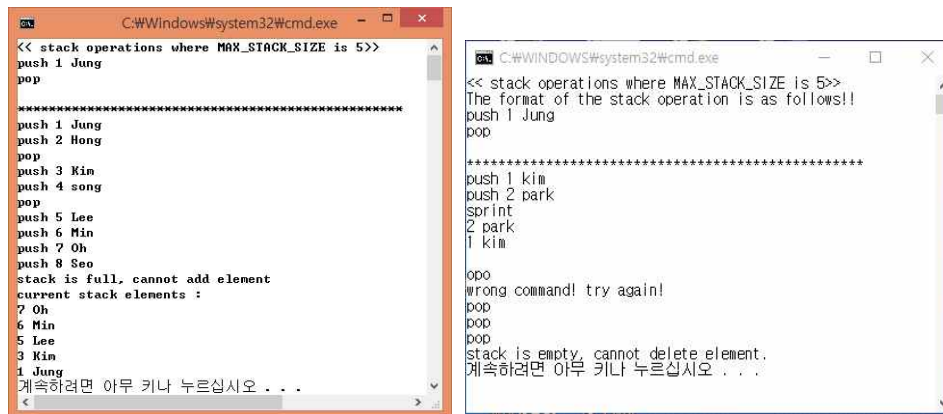
```
void stackFull()
{
    fprintf(stderr, "Stack is full, cannot add element");
    exit(EXIT_FAILURE);
}
```

**Program 3.3:** Stack full

[구현조건]

- ① 사용자입력으로부터 데이터 추출을 위해 gets\_s, strtok\_s, strcmp, sscanf\_s, strlen 등을 사용
  - ② push, pop 함수는 교재 코드를 수정 없이 그대로 사용할 것
  - ③ stackFull은 코드를 수정하되, pop()을 사용하여 현재 스택요소들을 출력해 준 후 exit(EXIT\_FAILURE)을 호출하도록 구현할 것
  - ④ stackEmpty는 새로 정의하되, 반환형은 임의의 에러키를 가지는 element형이어야 함
- ※ EXIT\_FAILURE를 사용하기 위해 stdlib.h를 인클루드해야 함

⑤ 잘못된 커맨드를 입력하면 에러메시지를 출력 후 다시 사용자 입력을 받도록 함  
[ 실행 예 ]



```
C:\Windows\system32\cmd.exe
<< stack operations where MAX_STACK_SIZE is 5>>
push 1 Jung
pop
*****
push 1 Jung
push 2 Hong
pop
push 3 Kim
push 4 song
pop
push 5 Lee
push 6 Min
push 7 Oh
push 8 Seo
stack is full, cannot add element
current stack elements :
7 Oh
6 Min
5 Lee
3 Kim
1 Jung
계속하려면 아무 키나 누르십시오 . . .

C:\WINDOWS\system32\cmd.exe
<< stack operations where MAX_STACK_SIZE is 5>>
The format of the stack operation is as follows!!
push 1 Jung
pop
*****
push 1 kim
push 2 park
sprint
2 park
1 kim
pop
wrong command! try again!
pop
pop
pop
stack is empty, cannot delete element.
계속하려면 아무 키나 누르십시오 . . .
```

2. [ 정적할당배열을 이용한 선형큐(linear queue) ] 다음과 같은 선형 큐를 생성하고 실행 예와 같이 수행되는 프로그램을 작성하라. 이를 위해, addq, deleteq, queueFull, queueEmpty qprint(queue의 내용을 출력) 함수를 구현하여야 한다.

[자료형과 함수의 정의]

```
#define MAX_QUEUE_SIZE 5
typedef struct {
    int id;                // unique id
    char name[MAX_NAME_SIZE]; //last name
} element;
element queue[MAX_QUEUE_SIZE];
int rear = -1;
int front = -1;
```

```
void addq(element item)
{ /* add an item to the queue */
    if (rear == MAX_QUEUE_SIZE-1)
        queueFull();
    queue[++rear] = item;
}
```

**Program 3.5: Add to a queue**

```
element deleteq()
{ /* remove element at the front of the queue */
    if (front == rear)
        return queueEmpty(); /* return an error key */
    return queue[++front];
}
```

**Program 3.6: Delete from a queue**

[ 구현 조건 ]

- ① 사용자입력으로부터 데이터 추출을 위해 gets\_s, strtok\_s, strcmp, sscanf\_s, strlen 등을 사용
- ② addq, deleteq 함수는 교재 코드를 수정 없이 그대로 사용할 것
- ③ queueFull은 아래와 같이 정의함
  - ✓ front == -1, rear == MAX\_QUEUE\_SIZE -1의 경우 아래와 같이 구현
  - “Queue is full, cannot add element!” 메시지를 출력
  - deleteq를 호출하여 현재 큐내용을 출력
  - exit(EXIT\_FAILURE) 호출
  - ✓ 그 이외에 대해서는 큐의 항목들을 이동
- ④ queueEmpty는 새로 정의하되, 반환형은 임의의 에러키를 가지는 element형이어야 함
- ⑤ 잘못된 커맨드를 입력하면 에러메시지를 출력 후 다시 사용자 입력을 받도록 함

[ 실행 예 ]

```
C:\WINDOWS\system32\cmd.exe
<< linear queue operations where MAX_QUEUE_SIZE is 3>>
add 1 Jung
delete
print
*****
add 1 kim
add 3 hong
add 5 kor
qprint
1 kim
3 hong
5 kor

delete
qprint
3 hong
5 kor

add 7 America
array shifting...
qprint
3 hong
5 kor
7 America

add 8 park
queue is full, cannot add element
current queue elements :
3 hong
5 kor
7 America
계속하려면 아무 키나 누르십시오 . . .
```

```
C:\WINDOWS\system32\cmd.exe
>
add 1 Jung
delete
print
*****
add 1 lee
add 3 seo
delete
qprint
3 seo

delete
delete
queue is empty, cannot delete element.
계속하려면 아무 키나 누르십시오 . . .
```

3. [ 환형큐(circular queue) ] 2번 문제의 프로그램을 환형큐 프로그램으로 수정하라. add, delete, qprint 명령어를 구현하시오.

[자료형과 함수의 정의]

```
typedef struct {
    int id;                // unique id
    char name[MAX_NAME_SIZE]; //last name
} element;
element *queue;
int capacity = 2;
int rear = 0;
int front = 0;

element deleteq()
/* remove front element from the queue */
{
    element item;
    if (front == rear)
        return queueEmpty(); /* return an error key */
    front = (front+1) % MAX_QUEUE_SIZE;
    return queue[front];
}
```

**Program 3.8:** Delete from a circular queue

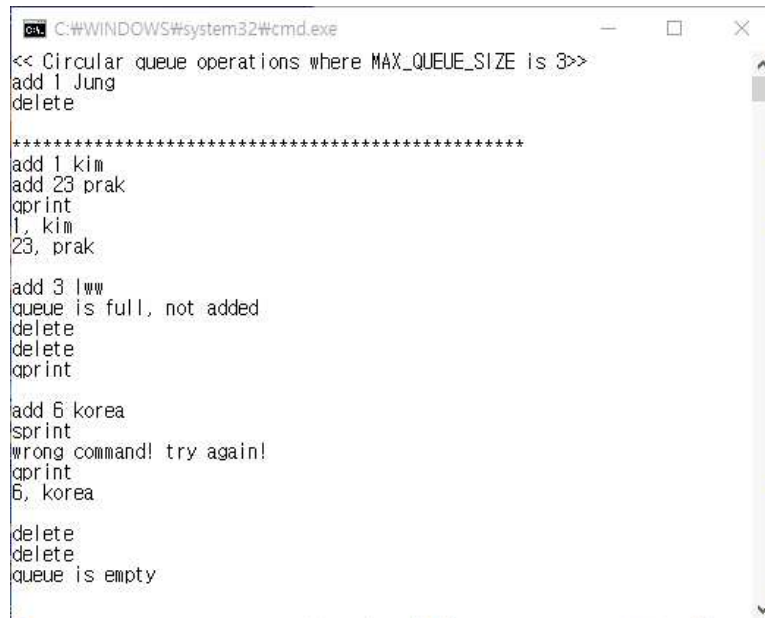
```
void addq(element item)
/* add an item to the queue */
{
    rear = (rear+1) % capacity;
    if (front == rear)
        queueFull();
    queue[rear] = item;
}
```

**Program 3.9:** Add to a circular queue

[ 구현 조건 ]

- ① 사용자입력으로부터 데이터 추출을 위해 `gets_s`, `strtok_s`, `strcmp`, `sscanf_s`, `strlen` 등을 사용
- ② 전역변수 `front`, `rear`의 초기값은 각각 0, 0
- ③ `addq`, `deleteq` 함수는 수정 없이 사용하기
- ④ `circular queue`를 전역변수 `element *queue;`로 선언
- ⑤ `MAX_QUEUE_SIZE` 3

## [ 실행 예 ]



```
C:\WINDOWS\system32\cmd.exe
<< Circular queue operations where MAX_QUEUE_SIZE is 3>>
add 1 Jung
delete

*****
add 1 kim
add 23 prak
qprint
1, kim
23, prak

add 3 lww
queue is full, not added
delete
delete
qprint

add 6 korea
sprint
wrong command! try again!
qprint
6, korea

delete
delete
queue is empty
```

### ■ 제출 형식

- 공학인증 시스템(ABEEK)에 과제를 올릴 때 제목:
  - 1차 제출: 학번\_이름\_DS\_05(1), 2차 제출: 학번\_이름\_DS\_05(2)
- 솔루션 이름 : DS\_05
- 프로젝트 이름 : 1, 2, 3
- 실행화면을 캡처하여 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\_05(1).zip, 2차 제출: 학번\_이름\_DS\_05(2).zip
- 제출은 2회 걸쳐 가능(수정 시간 기준으로 처리)