

## 자료구조응용

### 08. 동적배열 환형큐, 미로탐색 (10점)

2020.4.8. (수)

1. [ 동적할당배열을 이용한 환형큐 ] 자료구조응용 07. 2번 문제의 프로그램을 동적할당 배열을 이용한 환형큐 프로그램으로 수정하라. queueFull 함수는 queue capacity를 두 배로 확장하는 Program 3.10으로 구현한다. (5점)

[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 */

    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(); /* double capacity */
    queue[rear] = item;
}
```

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

---

```

void queueFull()
{
    int start;
    /* allocate an array with twice the capacity */
    element* newQueue;
    MALLOC(newQueue, 2 * capacity * sizeof(*queue));

    /* copy from queue to newQueue */
    start = (front+1) % capacity; rear--;
    if (start < 2)
        /* no wrap around */
        copy(queue+start, queue+start+capacity-1, newQueue);
    else
        /* queue wraps around */
        copy(queue+start, queue+capacity, newQueue);
        copy(queue, queue+rear+1, newQueue+capacity-start);
    }

    /* switch to newQueue */
    front = 2 * capacity - 1;
    rear = capacity - 1;
    capacity *= 2;
    free(queue);
    queue = newQueue;
}

```

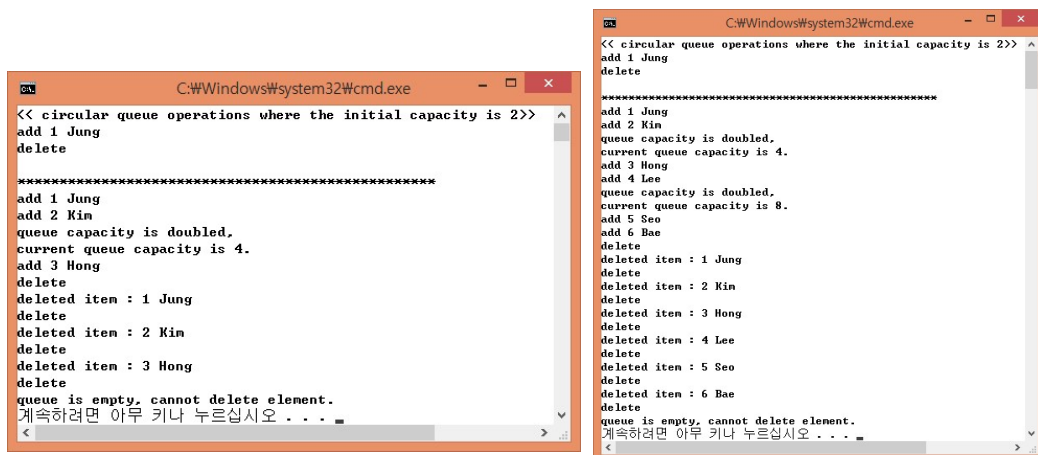
---

**Program 3.10:** Doubling queue capacity

[ 구현 조건 ]

- ① 사용자입력으로부터 데이터 추출을 위해 `gets`, `strtok`, `strcmp`, `sscanf`, `strlen` 등을 사용
- ② 전역변수 `capacity`, `front`, `rear`의 초기값은 각각 2, 0, 0
- ③ `addq`, `deleteq` 함수는 수정 없이 사용하기  
(단, `deleteq`의 `MAX_QUEUE_SIZE`를 `capacity`로 수정함)
- ④ `circular queue`를 전역변수 `element *queue;`로 선언
- ⑤ `main`에서 동적 할당으로 `capacity` 2의 초기 `queue`를 생성함
- ⑥ `copy` 함수를 직접 정의해야 함
- ⑦ 필요하다면 `queueFull` 함수의 아래 부분에 정보출력코드를 일부 추가할 수 있음
- ⑧ 실행창에서 `quit`을 입력하면 프로그램을 종료한다.

[ 실행 예 ]



## 2. Program 3.12 Maze search function을 사용하는 미로탐색 프로그램을 작성하여라.(5점)

### [프로그램 설명]

다음 ① ~ ⑤의 변수는 모두 전역으로 선언되며 정적할당을 사용함.

#### ① maze

- 입구는 left top, 출구는 right bottom으로 가정
- 미로 데이터입력 ("input.txt", 최대 10×10 행렬로 파일입력)

```
4 5
0 0 0 1 1
1 1 0 0 0
1 0 1 1 1
1 0 1 0 0
```

entrance (1, 1), exit (4, 5)

#### ② mark

#### ③ move

```
typedef struct {
    short int vert;
    short int horiz;
} offsets;
offsets move[8];
```

Name	Dir	move[dir].vert	move[dir].horiz
N	0	-1	0
NE	1	-1	1
E	2	0	1
SE	3	1	1
S	4	1	0
SW	5	1	-1
W	6	0	-1
NW	7	-1	-1

#### ④ stack

```
#define MAX_STACK_SIZE 100
typedef struct {
    short int row;
    short int col;
    short int dir;
} element;
element stack[MAX_STACK_SIZE];
int top = -1;
```

※자료구조응용 07. 1번 문제의 스택 소스를 활용할 것

#### ⑤ 기호상수 : #define TRUE 1

#define FALSE 0

전역변수 : int EXIT\_ROW, EXIT\_COL;

---

```

void path(void)
{
    /* output a path through the maze if such a path exists */
    int i, row, col, nextRow, nextCol, dir, found = FALSE;
    element position;
    mark[1][1] = 1; top = 0;
    stack[0].row = 1; stack[0].col = 1; stack[0].dir = 1;
    while (top > -1 && !found) {
        position = pop();
        row = position.row; col = position.col;
        dir = position.dir;
        while (dir < 8 && !found) {
            /* move in direction dir */
            nextRow = row + move[dir].vert;
            nextCol = col + move[dir].horiz;
            if (nextRow == EXIT_ROW && nextCol == EXIT_COL)
                found = TRUE;
            else if (!maze[nextRow][nextCol] &&
                ! mark[nextRow][nextCol]) {
                mark[nextRow][nextCol] = 1;
                position.row = nextRow; position.col = nextCol;
                position.dir = ++dir;
                push(position);
                row = nextRow; col = nextCol; dir = 0;
            }
            else ++dir;
        }
    }
    if (found) {
        printf("The path is:\n");
        printf("row col\n");
        for (i = 0; i <= top; i++)
            printf("%2d%5d", stack[i].row, stack[i].col);
        printf("%2d%5d\n", row, col);
        printf("%2d%5d\n", EXIT_ROW, EXIT_COL);
    }
    else printf("The maze does not have a path\n");
}

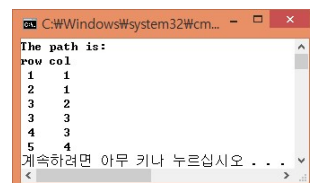
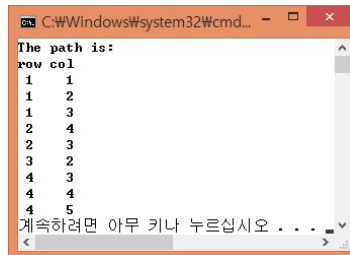
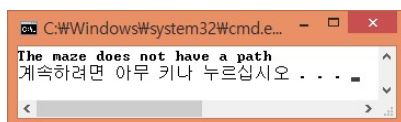
```

---

**Program 3.12:** Maze search function

### [실행 예]

4 5	4 5	5 4
0 0 0 1 1	0 0 0 1 1	0 1 1 1
1 1 0 0 0	1 1 0 0 0	0 1 1 0
1 0 1 1 1	1 0 1 1 1	1 0 0 1
1 0 1 0 0	1 0 0 0 0	1 1 0 1
		1 0 1 0



■ 제출 형식

- 솔루션 이름 : DS 08
- 프로젝트 이름 : 1, 2
- 각 소스파일에 주석처리  
“학번 이름”  
“본인은 이 소스파일을 복사 없이 직접 작성하였습니다.”
- 솔루션 정리 후 솔루션 폴더를 “학번.zip”으로 압축하여 과제 게시판에 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!
- 마감 : 4월 9일(목) 자정