자료구조

Homework #5

담당 교수	노서영 교수님
학과	도시공학과
학번	2018032027
이름	황슬비

자료구조

a. Windows + GNU GCC + Eclipse 환경 선택

b. 소스코드 및 설명

CircularQ.c

```
/*
 * circularQ.c
 *
 *  Data Structures, Homework #5
 *  Department of Computer Science at Chungbuk National University
 */

#include <stdio.h>
#include <stdlib.h>

#define MAX_QUEUE_SIZE 4

typedef char element;
typedef struct { //queue 구성요소 구조체로 정의
	element queue[MAX_QUEUE_SIZE];
	int front, rear;
}QueueType;


QueueType *createQueue();
int freeQueue(QueueType *cQ);
int isEmpty(QueueType *cQ);
int isFull(QueueType *cQ);
int enQueue(QueueType *cQ, element item);
int deQueue(QueueType *cQ, element* item);
void printQ(QueueType *cQ);
void debugQ(QueueType *cQ);
element getElement();

int main(void)
{



	QueueType *cQ = createQueue();
	element data;
	char command;

	do{
		printf("[----- [황슬비] [2018032027] -----]");
		printf("\n-----------------------------------------------------\n");
		printf("                    Circular Q                   \n");
		printf("-----------------------------------------------------\n");
```

```
                  printf(" Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q \n");
                  printf("------------------------------------------------------\n");

                  printf("Command = ");
                  fflush(stdout);
                  scanf(" %c", &command);

                  switch(command) {
                  case 'i': case 'I':
                           data = getElement(); //큐에 넣을 값 입력
                           enQueue(cQ, data); //큐에 삽입
                           break;
                  case 'd': case 'D':
                           deQueue(cQ, &data); //큐에서 삭제
                           break;
                  case 'p': case 'P':
                           printQ(cQ);
                           break;
                  case 'b': case 'B':
                           debugQ(cQ);
                           break;
                  case 'q': case 'Q':
                 freeQueue(cQ);
                           break;
                  default:
                           printf("\n       >>>>>   Concentration!!   <<<<<     \n");
                           break;
                  }

         }while(command != 'q' && command != 'Q');


         return 1;
}

QueueType *createQueue() //큐 생성
{
         QueueType *cQ;
         cQ = (QueueType *)malloc(sizeof(QueueType));
         cQ->front = 0;
         cQ->rear = 0;
         return cQ;
}

int freeQueue(QueueType *cQ) //큐 할당 해제
```

```
{
    if(cQ == NULL) return 1;
    free(cQ);
    return 1;
}

element getElement()
{
         element item;
         printf("Input element = ");
         fflush(stdout);
         scanf(" %c", &item);



         return item;
}


/* complete the function */
int isEmpty(QueueType *cQ)
{

                  if(cQ->front==cQ->rear){ //front와 rear값이 같으면 1반환 (true)
                           return 1;
                  }
    return 0;
}

/* complete the function */
int isFull(QueueType *cQ)
{

         if(cQ->front==cQ->rear){//rear+1한 후, front와 rear가 같으면 full, 1반환
                  return 1;
                           }
   return 0;
}


/* complete the function */
int enQueue(QueueType *cQ, element item)
{
         cQ->rear = (cQ->rear+1)%MAX_QUEUE_SIZE; //rear +1, 나머지연산
         int ls;
```

```
        ls=isFull(cQ);
        if(ls==1){
                printf("Queue is Full!\n");
        }
        else{ //full이 아닐 때

        cQ->queue[cQ->rear] = item;//큐에 데이터 삽입
        }
        return 0;
}

/* complete the function */
int deQueue(QueueType *cQ, element *item)
{
        int ls;
        ls=isEmpty(cQ);
        if(ls==1){ //큐가 비었는지
                printf("Queue is Empty!\n");
        }
        else{
                return  cQ->queue[cQ->front=(cQ->front+1)%MAX_QUEUE_SIZE];  //front  한칸
증가시켜 값 삭제, front는 빈 곳를 가리키게 됨
        }
        return 0;
}


void printQ(QueueType *cQ)
{
        int i, first, last;

        first = (cQ->front + 1)%MAX_QUEUE_SIZE;
        last = (cQ->rear + 1)%MAX_QUEUE_SIZE;

        printf("Circular Queue : [");

        i = first;
        while(i != last){
                printf("%3c", cQ->queue[i]);
                i = (i+1)%MAX_QUEUE_SIZE;

        }
        printf(" ]\n");
}
```

```
void debugQ(QueueType *cQ)
{

	printf("\n---DEBUG\n");
	for(int i = 0; i < MAX_QUEUE_SIZE; i++)
	{
		if(i == cQ->front) {
			printf("  [%d] = front\n", i);
			continue;
		}
		printf("  [%d] = %c\n", i, cQ->queue[i]);

	}
	printf("front = %d, rear = %d\n", cQ->front, cQ->rear);
}
```

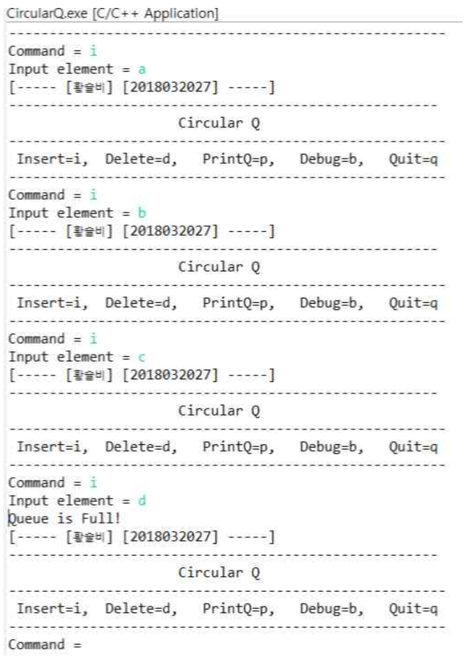
c. Github URL

https://github.com/seulbih/datastructure-hw5/blob/29d721585c1a335598fe3707e2f786bf109eb1ed/circularQ.c

자료구조

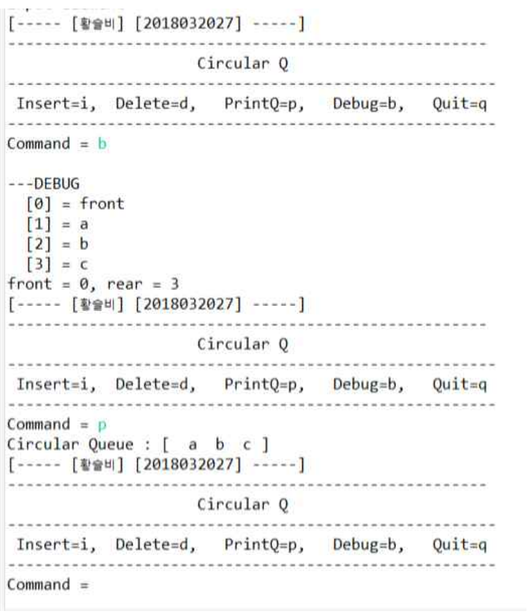
d.실행결과

insert ‘a’,‘b’,‘c’ / Full메세지 출력



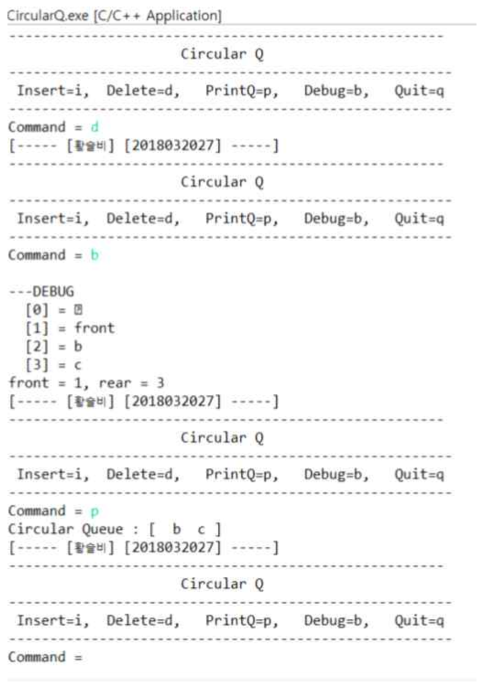
```
CircularQ.exe [C/C++ Application]
--------------------------------------------------
Command = i
Input element = a
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = i
Input element = b
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = i
Input element = c
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = i
Input element = d
Queue is Full!
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command =
```

Debug&Print



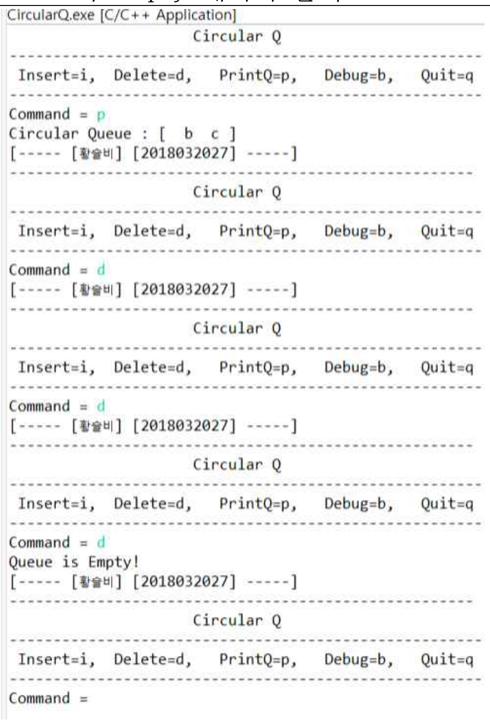
```
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = b

---DEBUG
  [0] = front
  [1] = a
  [2] = b
  [3] = c
front = 0, rear = 3
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = p
Circular Queue : [  a  b  c ]
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command =
```

Delete&Debug&Print



```
CircularQ.exe [C/C++ Application]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = d
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = b

---DEBUG
  [0] = 
  [1] = front
  [2] = b
  [3] = c
front = 1, rear = 3
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = p
Circular Queue : [  b  c ]
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command =
```

Delete /Empty 메시지 출력



```
CircularQ.exe [C/C++ Application]
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = p
Circular Queue : [  b  c ]
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = d
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = d
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command = d
Queue is Empty!
[----- [황슬비] [2018032027] -----]
--------------------------------------------------
                  Circular Q
--------------------------------------------------
 Insert=i,  Delete=d,   PrintQ=p,   Debug=b,   Quit=q
--------------------------------------------------
Command =
```

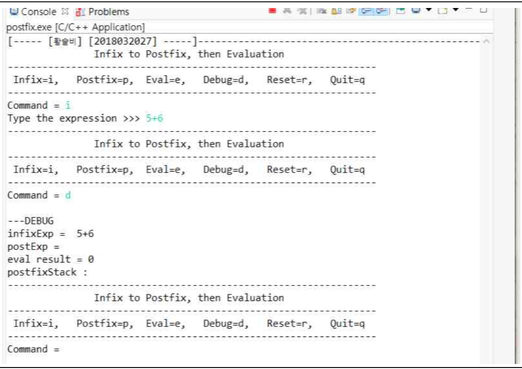
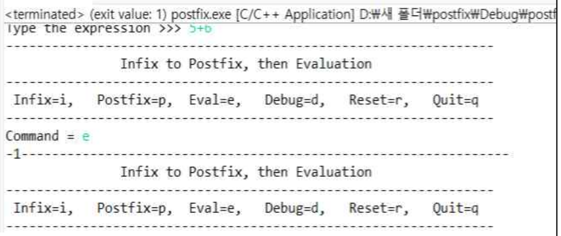
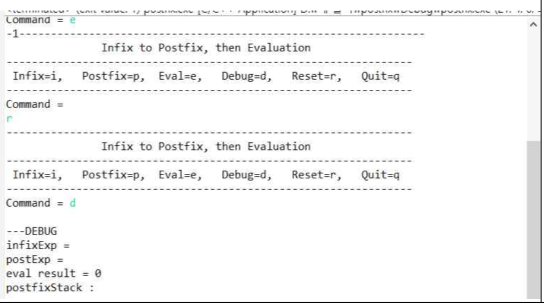
e. 소스코드 및 설명

postfix.c

```
/*
 * postfix.c
 *
 *  Data Structures, Homework #5
 *  Department of Computer Science at Chungbuk National University
 */
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define MAX_STACK_SIZE 10
#define MAX_EXPRESSION_SIZE 20
/* stack 내에서 우선순위, lparen = 0 가장 낮음 */
typedef enum{
        lparen = 0,  /* ( 왼쪽괄호 */
        rparen = 9,  /* ) 오른쪽괄호*/
        times = 7,   /* * 곱셈 */
        divide = 6,  /* / 나눗셈 */
        plus = 5,    /* + 덧셈 */
        minus = 4,   /* - 뺄셈 */
        operand = 1 /* 피연산자 */
} precedence;
char infixExp[MAX_EXPRESSION_SIZE];         /* infix expression을 저장하는 배열 */
char postfixExp[MAX_EXPRESSION_SIZE];       /* postfix로 변경된 문자열을 저장하는 배열 */
char postfixStack[MAX_STACK_SIZE]; /* postfix로 변환을 위해 필요한 스택 */
int evalStack[MAX_STACK_SIZE];              /* 계산을 위해 필요한 스택 */
int postfixStackTop = -1;  /* postfixStack용 top */
int evalStackTop = -1;       /* evalStack용 top */
int evalResult = 0;          /* 계산 결과를 저장 */
void postfixpush(char x);
char postfixPop();
void evalPush(int x);
int evalPop();
void getInfix();
precedence getToken(char symbol);
precedence getPriority(char x);
void charCat(char* c);
void toPostfix();
void debug();
void reset();
void evaluation();
int main()
{       char command;
        printf("[----- [황슬비] [2018032027] -----]");
        do{

printf("-----------------------------------------------------------------\n");
                printf("                                Infix  to  Postfix,  then  Evaluation
\n");

printf("-----------------------------------------------------------------\n");
                printf(" Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
\n");

printf("-----------------------------------------------------------------\n");
                printf("Command = ");
                fflush(stdout);
                scanf(" %c", &command);
                switch(command) {
                case 'i': case 'I':
                        getInfix();
                        break;
                case 'p': case 'P':
                        toPostfix();
                        break;
                case 'e': case 'E':
                        evaluation();
                        break;
                case 'd': case 'D':
                        debug();
                        break;
                case 'r': case 'R':
                        reset();
                        break;
                case 'q': case 'Q':
                        break;
                default:
                        printf("\n      >>>>>   Concentration!!   <<<<<     \n");
                        break;
                }
        }while(command != 'q' && command != 'Q');
        return 1;
}void postfixPush(char x) //연산자 push
{    postfixStack[++postfixStackTop] = x;
}char postfixPop()//
{    char x;
    if(postfixStackTop == -1) //empty
        return '\0';
    else {
         x = postfixStack[postfixStackTop--]; //pop하고 top하나 -
    }
    return x;
}void evalPush(int x)
{    evalStack[++evalStackTop] = x;
}int evalPop()
{    if(evalStackTop == -1)
        return -1;
    else
        return evalStack[evalStackTop--];
}/**
 * infix expression을 입력받는다.
 * infixExp에는 입력된 값을 저장한다.
 */
void getInfix()
{    printf("Type the expression >>> ");
    fflush(stdout);
    scanf("%s",infixExp);
}precedence getToken(char symbol)
{        switch(symbol) {
         case '(' : return lparen; //0
         case ')' : return rparen; //9
         case '+' : return plus; //7
         case '-' : return minus; //6
         case '/' : return divide; //5
         case '*' : return times; //4
         default : return operand; //1
         }
}precedence getPriority(char x)
{        return getToken(x);
}/**
 * 문자하나를 전달받아, postfixExp에 추가
 */
void charCat(char* c)
{        if (postfixExp == '\0') //postfixExp : postfix로 변경된 문자열 저장
                  strncpy(postfixExp, c, 1); //c에서 1개 문자열을 postfixExp로 복사
         else
                  strncat(postfixExp, c, 1);//c의 1개 문자열을 postfixExp뒤에 붙임
}/**
 * infixExp의 문자를 하나씩 읽어가면서 stack을 이용하여 postfix로 변경한다.
 * 변경된 postfix는 postFixExp에 저장된다.
 */
void toPostfix()
{        /* infixExp의 문자 하나씩을 읽기위한 포인터 */
         char *exp = infixExp;
	char x; /* 문자하나를 임시로 저장하기 위한 변수� */
	/* exp를 증가시켜가면서, 문자를 읽고 postfix로 변경 */
	while(*exp != '\0') //마지막전까지 loop
	{
		if(*exp=='1'){ //피연산자인 경우
			printf("%c",*exp); //바로 출력
		}
		else if (*exp=='9'){ //')'경우'
			while(postfixStack[postfixStackTop] !=0){//'('가 아니면
				x=postfixPop();
				printf("%c",x);//출력, ')'가 나올 때까지 반복
				postfixPop();
			}
		}
		else{
			while(getPriority(postfixStack[postfixStackTop])>=*exp){//스택안
에 우선순위가 높거나 같은게 있을 때
				x=postfixPop();
				printf("%c",x); //우선순위 높은것 뺌
				postfixPush(*exp); //스택에 푸쉬
			}
		}
	}
	while(*exp !='\0') //남은거 다 뽑아냄
		printf("%c", *exp);
}void debug()
{	printf("\n---DEBUG\n");
	printf("infixExp =  %s\n", infixExp);
	printf("postExp =  %s\n", postfixExp);
	printf("eval result = %d\n", evalResult);
	printf("postfixStack : ");
	for(int i = 0; i < MAX_STACK_SIZE; i++)
		printf("%c  ", postfixStack[i]);
	printf("\n");
}void reset()
{	infixExp[0] = '\0';
	postfixExp[0] = '\0';
	for(int i = 0; i < MAX_STACK_SIZE; i++)
		postfixStack[i] = '\0';
	postfixStackTop = -1;
	evalStackTop = -1;
	evalResult = 0;
}void evaluation()
{	/* postfixExp, evalStack�� �댁슜�� 怨꾩궛 */
	int x,y,z;
```

```
        char *exp=postfixExp;
        while(*exp !='\0'){
                if(*exp=1){ //피연산자인경우
                        evalPush(*exp); //eval에 넣어줌
                }
                else if(*exp==0 || *exp>1) { //연산자인경우
                        x = evalpop();
                        y = evalpop();
                        switch(*exp){
                                case '4' : z=y+x; break;
                                case '5' : z=y-x; break;
                                case '6' : z=y/x; break;
                                case '7' : x*y; break;
                        }
                        evalpush(z);
                        }
                }
        printf("%d\n", evalPop());
}
```

f. 실행결과

infix&Debug	postfix&eval
	
reset&Debug	
	

```
Console ☒  Problems
postfix.exe [C/C++ Application]
[----- [황슬비] [2018032027] -----]------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
Command = i
Type the expression >>> 5+6
------------------------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
Command = d

---DEBUG
infixExp =  5+6
postExp =
eval result = 0
postfixStack :
------------------------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
Command =
```

```
<terminated> (exit value: 1) postfix.exe [C/C++ Application] D:₩새 폴더₩postfix₩Debug₩postf
Type the expression >>> 5+6
------------------------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
Command = e
-1-----------------------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
```

```
Command = e
-1-----------------------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
Command =
r
------------------------------------------------------------------
                Infix to Postfix, then Evaluation
------------------------------------------------------------------
 Infix=i,   Postfix=p,  Eval=e,   Debug=d,   Reset=r,   Quit=q
------------------------------------------------------------------
Command = d

---DEBUG
infixExp =
postExp =
eval result = 0
postfixStack :
```

g. GitHub URL
https://github.com/seulbih/datastructure-hw5/blob/29d721585c1a335598fe3707e2f786bf109eb1ed/postfix.c