**3rd Assignment**

컴퓨터전공

2013011491

안찬영

Is\_empty 함수부터 시작하면,

int is\_empty(QueueType \*q)

{if(q->front==q->rear){

return 1;

}else{

return 0;}}

is\_empty의 함수는 큐가 비어 있을 경우 1을 반환 아닐 경우 0을 반환 한다. 큐의 front와 rear가 같은 위치를 가리킬 경우에 그 큐는 빈상태가 된다 그러므로 같을 경우 TRUE값을 반환하고 아닐 경우 FALSE 값을 반환한다.

int is\_full (QueueType \*q)

{if((q->front) - (q->rear) == (-1)\*(MAX\_QUEUE\_SIZE-1) || (q->front) - (q->rear) == 1){

return 1;

}else

return 0;}

is\_full함수는 큐가 가득 차 있을 경우 1을 반환, 아닐 경우 0을 반환한다. Front의 위치가 rear보다 한 칸 뒤를 가리킬 경우에 그 큐는 full상태가 된다. 그렇지만 rear가 저장 배열의 마지막을 가리키고 있을 경우에 front는 저장 배열의 맨 앞을 가리키고 있으므로, 그 차이는 큐최대사이즈-1이다. 그러므로 front와 rear의 차이가 1이거나, 큐최대사이즈-1 일 경우 TRUE값을 반환하고, 아닐 경우 FALSE 값을 반환한다.

void enqueue( QueueType \*q, element item )

{if(is\_full(q)){

error("Queue is full\n");

}else if(q->rear != MAX\_QUEUE\_SIZE-1){

q->rear++;} else{q->rear =0;}

q->queue[q->rear]=item;

printf("enqueue() = %d\n",item);

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

enqueue함수는 먼저 큐가 가득 찼는지 검사하고, 아닐 경우에 rear를 하나 증가시키고 rear가 가리키는 위치에 자료를 저장한다. 여기서 고려할 것은 rear가 가리키는 위치가 배열의 마지막일 경우에는 0으로 회전시켜 주어야 한다. Rear의 위치를 증가 시킨 다음에 큐의 배열에 데이터를 저장하고 저장정보를 출력한다.

element dequeue(QueueType \*q){

if(is\_empty(q)){

error("Queue is empty\n");}else if(q->front != MAX\_QUEUE\_SIZE-1){

q->front++; }else{

q->front=0;

}

printf("dequeue() = %d\n",q->queue[q->front]);

printf("front = %d, rear = %d\n",q->front,q->rear);

return q->queue[q->front];}

dequeue함수는 함수가 비어 있는지 확인한 후, 그렇지 않을 경우 front가 가리키는 위치를 증가 시킨 뒤 출력한다. 여기서는 enqueue함수와 유사하게 front가 가리키는 위치가 저장 배열의 마지막일 경우 front를 0으로 회전시켜 준다. Front의 위치를 증가 시킨 다음에는 삭제정보를 출력하고 삭제될 데이터를 출력한다.

element peek(QueueType \*q)

{

if(is\_empty){

error("Queue is empty\n");

}else{

printf("%d\n",q->queue[q->front+1]);

return q->queue[q->front+1];

}}

peek함수는 큐가 비어 있는지 확인 한 후, 비어 있지 않을 경우, 큐의 맨 앞 데이터를 출력한다.

int main(int argc, char \*argv[])

{

char command;

int key;

FILE\*input=fopen(argv[1],"r");

QueueType q;

init(&q);

printf("init front =0, rear=0\n");

while(1){

command = fgetc(input);

if(feof(input)) break;

switch(command){

case 'e' :

fscanf(input,"%d",&key);

enqueue(&q,key);

break;

case 'd' :

key=dequeue(&q);

break;

case 'p' :

key=peek(&q);

break;

}}

fclose(input);

return 0;}

main 함수에서는 먼저 파일을 열고, 큐를 생성후에 초기화 한다. 그 다음 파일로부터 파일 끝에 도달할 때까지 반복문으로 문자를 입력 받는다. 입력 받는 문자에 따라 스위치문을 통해 행해지는 연산이 다르다. E를 입력 받을 경우, enqueue함수를 실행하고 d를 입력 받을 경우 dequeue 함수 p를 입력 받을 경우 peek함수를 실행한다. 마지막으로 파일을 종료하고 0값을 반환 한 뒤 종료한다.