Data Structures

Lab # 05

Data & Knowledge Engineering Laboratory
Kyung Hee University, Korea

Chapter 5 Programming Exercise

소스 코드

- **샘플 모드 중 아래 경로에 위치한 2개의 파일을 수정**
  - 사용할 샘플 파일 : QueType.h, QueType.cpp

- **실습간 사용할 큐의 Item은 정수(Integer)로 사용할 것**

QueType.h 중 클래스 선언부

```
typedef char ItemType;
class QueType
{
public:
    QueType();
    QueType(int max);
    ~QueType();
    void MakeEmpty();
    bool IsEmpty() const;
    bool IsFull() const;
    void Enqueue(ItemType newItem);
    void Dequeue(ItemType& item);
private:
    int front;
    int rear;
    ItemType* items;
    int maxQue;
};
```

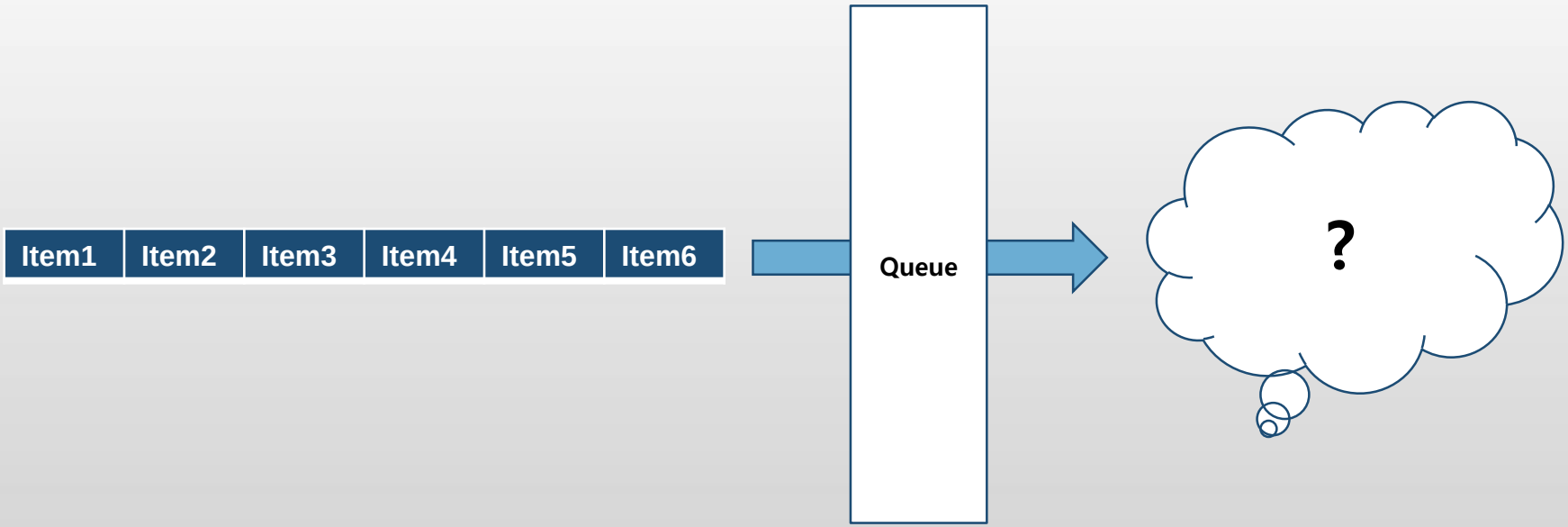
QueType.h 중 클래스 선언부

```
typedef int ItemType;
class QueType
{
public:
    QueType();
    QueType(int max);
    ~QueType();
    void MakeEmpty();
    bool IsEmpty() const;
    bool IsFull() const;
    void Enqueue(ItemType newItem);
    void Dequeue(ItemType& item);
private:
    int front;
    int rear;
    ItemType* items;
    int maxQue;
};
```

실습 내용1

■ 문제

- ❖ 큐의 동작 방법을(Enqueue, Dequeue) 익힌다.
- ❖ 10개의 아이템을 랜덤한 값을 가지도록 생성
- ❖ Enqueue를 통해 아이템의 값을 저장한다. 큐에 아이템을 넣기 전에 아이템 값을 출력
- ❖ Dequeue를 통해 값을 가져와 출력한다.

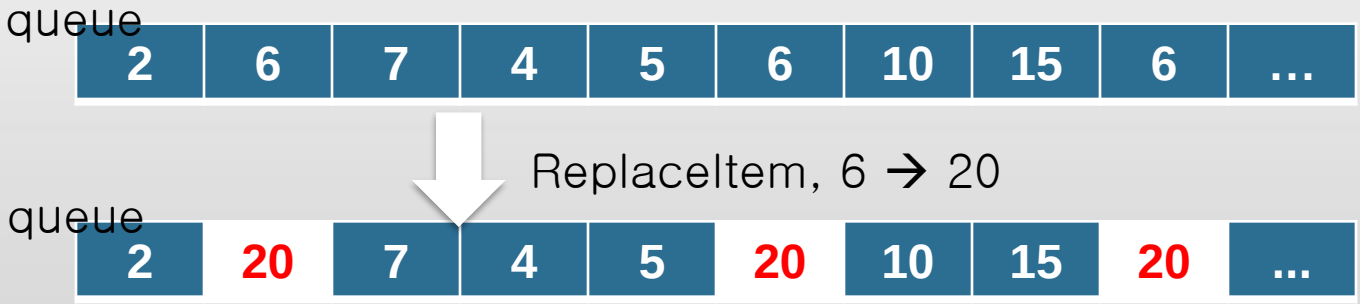


실습 내용2

ReplaceItem이란 함수를 구현하라.

- ReplaceItem은 내부에 존재하는 값을 새로운 값으로 치환함
- **A. Client 함수로 작성**
  - **프로토타입 :** void ReplaceItem(QueType<int> &queue, int oldItem, int newItem);
- **b. 멤버함수로 작성**
  - **프로토타입 : void ReplaceItem(ItemType oldItem, ItemType newItem);**

예 제



실습 내용2(help)

a. 클라이언트 함수

```
ReplaceItem(...)
{
   임시 변수, 큐 선언

   while(!queue.IsEmpty())
   { Dequeue(...);
      if(oldItem과 같은가)
        임시큐.Enqueue(새 값)
      else
        임시큐.Enqueue(기존값)
   }
   while(!임시큐.IsEmpty())
   { 임시큐.Dequeue(...)
      queue.Enqueue(...)
   }
```

b. 멤버 함수

```
template<class ItemType>
void QueType<ItemType>::ReplaceItem(…)
{
   … // 메서드의 장점을 이용해 구현한다.
   // 변수를 직접적으로 엑세스하여 검사.
}
```

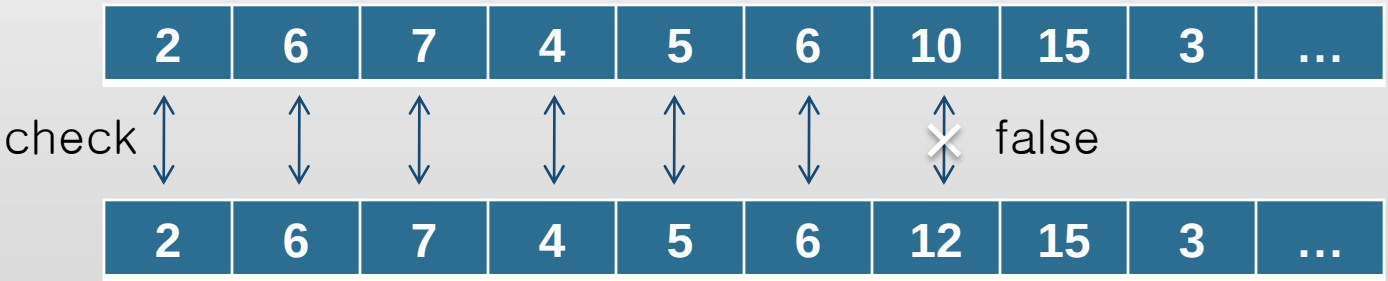
실습 내용3

■ 문 제

❖ 두 개의 큐가 같은지 검사하는 함수를 구현하시오. (문제 참고) 만약 두 개의 큐가 같다면 TRUE를 리턴하고 다르다면 FALSE를 리턴하시오.

- **a. Client 함수로 작성**
  - **프로토타입 : bool Identical(QueType<int> queue1, QueType<int> queue2)**
- **b. 멤버함수로 작성**
  - **프로토타입 : bool Identical(Queue<ItemType> queue);**

■ 예 제



❖ 순차적으로 비교하며 같은지 검사함

실습 내용4

■ 문 제

❖ 큐에 몇 개의 아이템이 저장되었는지를 리턴하는 int형의 함수를 작성하라

- **a. Client 함수로 작성**
  - **프로토타입 :** int Length(QueType<int> queue);
- **b. 멤버함수로 작성**
  - 프로토타입 : int Length();

■ 예 제

실습 내용4(help)

a. 클라이언트 함수

```
int Length(QueType<int> &queue)
{
    ... // 임시큐를 선언합니다.
    // Dequeue를 하면서 카운트를 합니다.
    // 카운트를 한 뒤 값을 임시 큐에 저장합니다.
    // 큐를 복원합니다.
}
```

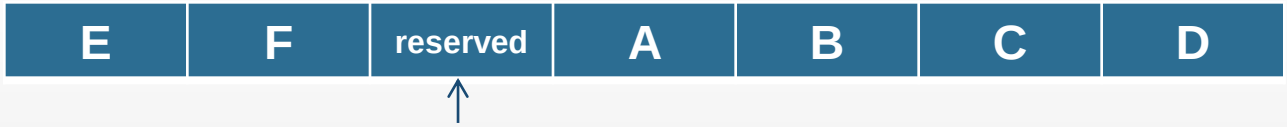
b. 멤버 함수

- 멤버변수인 Front 와 rear를 이용하여 아이템의 개수를 계산함

실습 내용5

■ 문 제

- ❖ 현재 구현된 큐 자료구조는 가득찬 경우와 비어있는 경우를 구분하기 위하여 Front 앞을 비워두고 예약 공간으로 사용하였다.
- ❖ 기존 큐가 Full 상태인 경우



Front 앞을 비워놓기 때문에 사용하지 못하는 공간

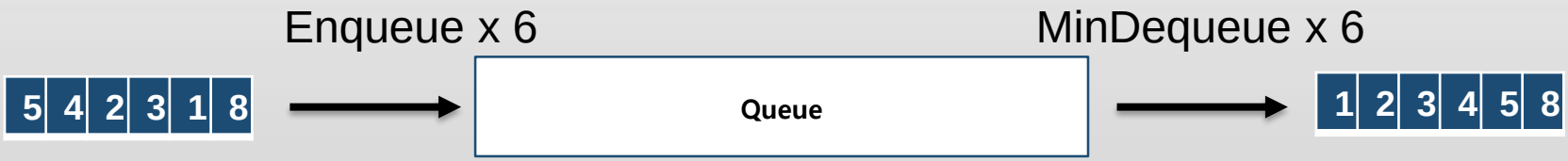
- ❖ Front 앞을 사용하도록 큐를 재 설계를 하고, 구현하여라
  - Int 타입의 Length 변수를 클래스의 private로 추가
  - Length 변수를 이용하여 큐의 크기를 계산함
  - 필요하다면 멤버 함수를 변경하시오
    - Etc) Enqueue, dequeue, MakeEmpty, 생성자, isEmpty, isFull

실습 내용6

- **Queue의 Dequeue동작을 수정하시오**

- **기존의 Dequeue는 '넣은 순서'를 기준으로 먼저 넣은 값을 먼저 나오도록 구현되어 있다.**

- **수정할 Dequeue는 아이템의 '값이 작은 순서'를 기준으로 아이템이 반환되도록 한다.**
  - ❖ MinDequeue함수를 선언하고 수정된 내용을 이 함수에 작성한다.



실습 내용6(help)

- **Int minimum_pos를 멤버 변수로 선언한다**
  - ❖ 이 변수는 Queue내부 배열에서 가장 작은 값을 가지는 위치를 표시한다

- **MinDequeue가 수행되면 값을 반환하고, 이 자리는 -1 값으로 초기화된다.**

- **Enqueue가 수행되면, -1로 초기화 되어 있는 위치에 입력 받은 값을 삽입한다.**

- **Enqueue 혹은 MinDequeue가 수행되면, minimum_pos가 자신의 위치를 찾도록 초기화한다.**