Data Structures

Lab # 05

Data & Knowledge Engineering Laboratory
Kyung Hee University, Korea

Chapter 5 Programming Exercise

소스 코드

- **아래 경로에 위치한 2개의 파일을 수정해 사용하라**
  - ₩labplus₩Lab, c++ 3rd₩Chapter4₩queue₩QueType.h, QueType.cpp

- **실습에서 큐의 ItemType은 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;
};
```

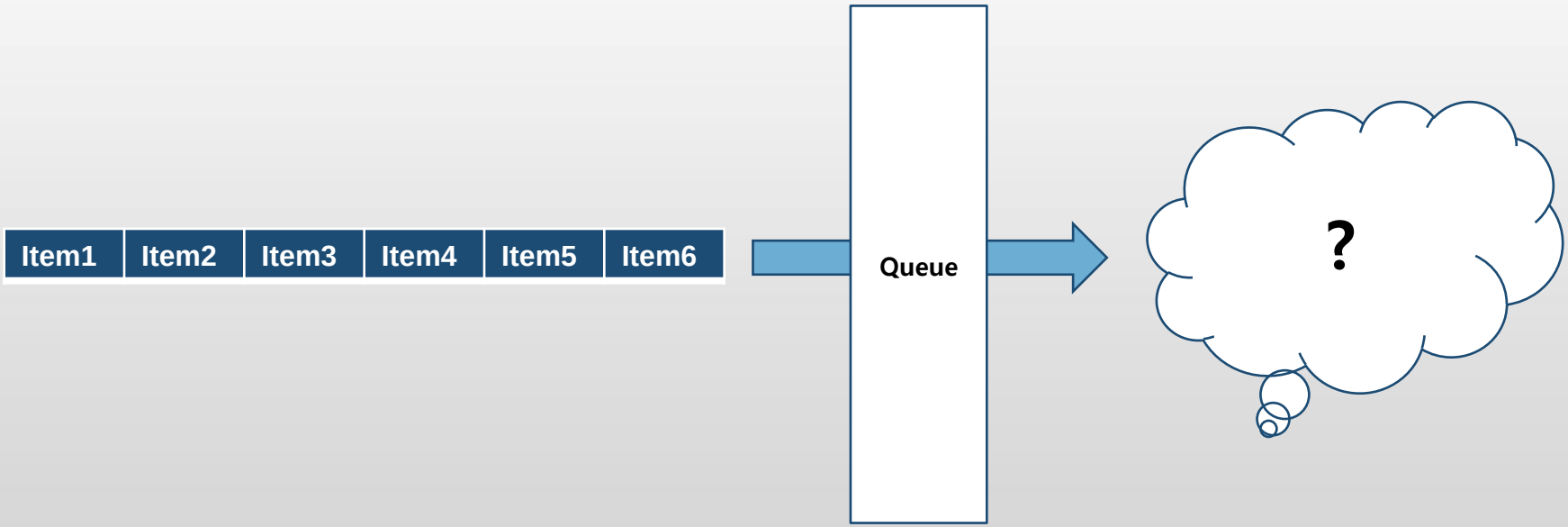
QueType.h 중 클래스 선언부

```
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.
- ❖ 랜덤 값을 갖는 아이템 10개 생성
- ❖ Enqueue로 아이템을 큐에 저장한다. 큐에 아이템을 넣기 전에 아이템을 출력한다
- ❖ Dequeue로 큐에 저장된 아이템을 가져와 출력한다.



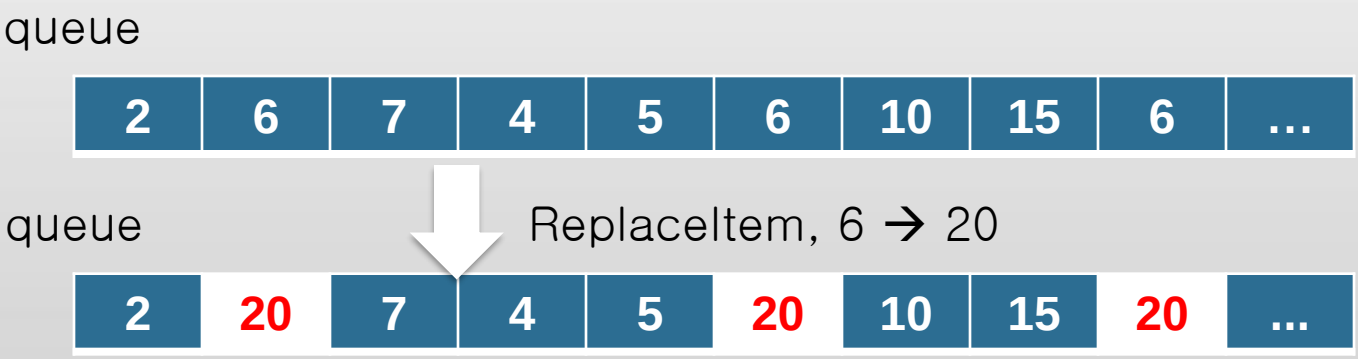
실습 내용2

■ ReplaceItem 함수를 구현하라.

❖ ReplaceItem은 큐 내부에 존재하는 어떤 값을 새로운 값으로 바꾸는 함수

- **A. Client 함수로 작성**
  - **프로토타입 : void ReplaceItem(QueType& queue, int oldItem, int newItem);**
- **b. 멤버함수로 작성**
  - **프로토타입 : void ReplaceItem(ItemType oldItem, ItemType newItem);**

■ 예 제



실습 내용2(help)

a. 클라이언트 함수

```
ReplaceItem(QueType& queue, int oldItem, int newItem)
{
   임시 큐 선언

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

b. 멤버 함수

```
void QueType::ReplaceItem(ItemType oldItem, ItemType newItem)
{
  // 멤버 함수의 장점을 이용해 구현한다.
  // 멤버 변수에 직접 접근해 검사.
}
```

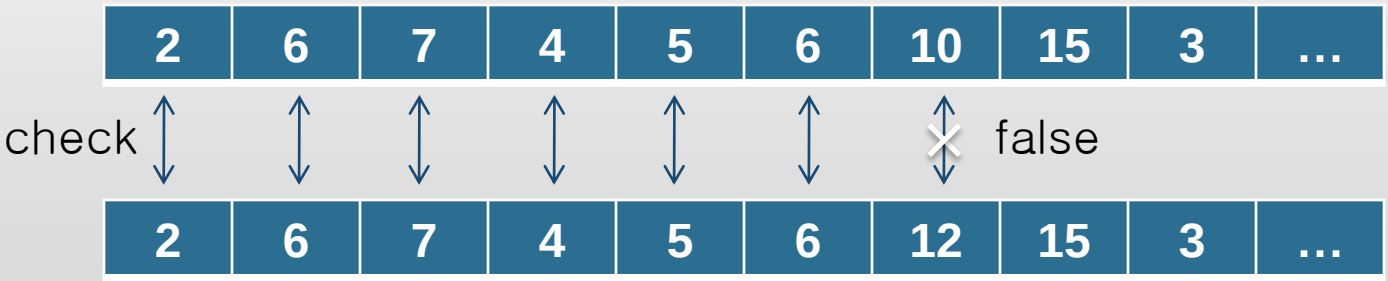
실습 내용3

■ 문 제

❖ 두 개의 큐가 같은지 검사하는 함수를 구현하라.
만약 두 개의 큐가 같다면 TRUE를 리턴하고 다르다면 FALSE를 리턴하라.

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

■ 예 제



❖ 순차적으로 두 값을 검사함

실습 내용4

■ 문 제

❖ 큐에 저장된 아이템 수를 알려주는 함수를 구현하라

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

■ 예 제

실습 내용4(help)

a. 클라이언트 함수

```
int Length(QueType& queue)
{
    ... // 임시큐를 선언합니다.
    // Dequeue를 하면서 아이템의 수를 셉니다.
    // Dequeue한 아이템 값을 임시 큐에 저장합니다.
    // 임시 큐를 사용해 큐를 복원합니다.
}
```

b. 멤버 함수

- 멤버변수인 Front 와 rear를 이용하여 아이템의 개수를 계산함

실습 내용5

■ 문 제

- ❖ 현재 구현된 큐 자료구조는 가득 찬 경우와 빈 경우를 구분하기 위해 Front가 위치한 공간을 예약 공간으로 사용하였다.
- ❖ 기존 큐가 Full 상태인 경우

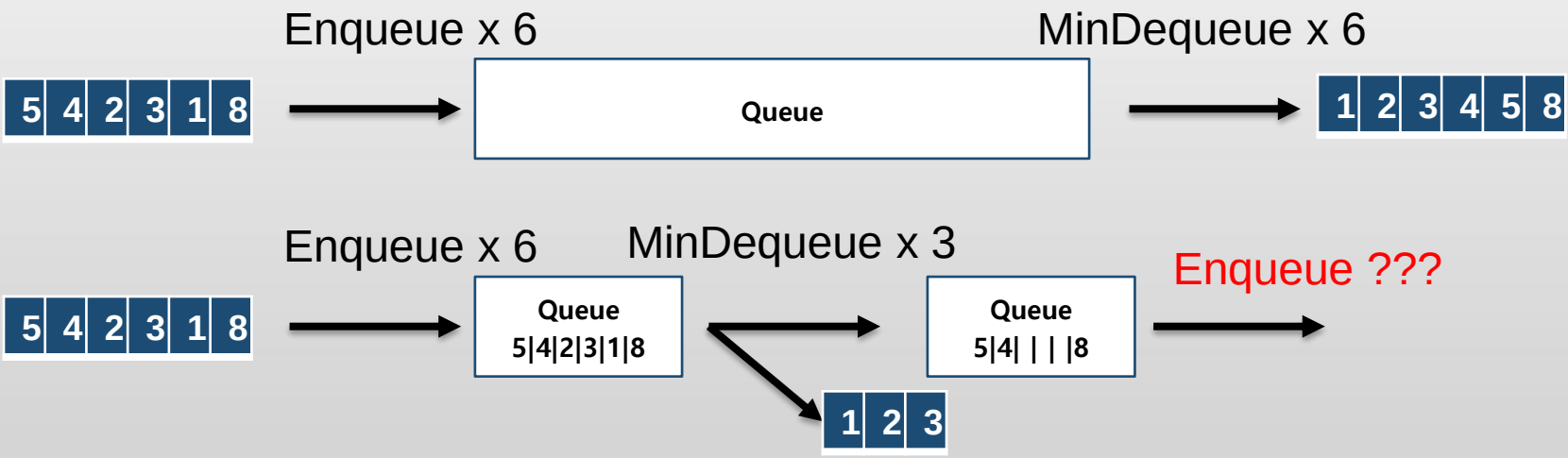
- ❖ 예약 공간을 사용할 수 있는 큐를 구현하여라
  - Int 타입의 Length 변수를 클래스의 멤버 변수로 추가
  - Length 변수를 이용하여 큐의 크기를 계산함
  - 변경한 큐에 맞게 멤버 함수를 수정하여라

실습 내용6

■ 문 제

- ❖ '값이 작은 순서'를 기준으로 아이템이 반환되는 함수 MinDequeue를 구현하라.
- ❖ MinDequeue로 인해 생기는 문제를 해결하기 위해 Enqueue 함수를 수정하라
- ❖ 실습 5에서 구현한 큐를 사용하라.
  - **멤버함수로 작성**
    - **프로토타입 : void MinDequeue(ItemType& item);**

■ 예 제



실습 내용6(help)

- **Int minimum_pos를 멤버 변수로 선언한다**
  - ❖ 이 변수는 Queue에서 가장 작은 아이템의 위치를 표시한다

- **MinDequeue가 실행되면 가장 작은 아이템을 반환하고, 그 자리는 -1로 초기화된다.**

- **Enqueue가 실행되면, 먼저 -1이 있는 위치에 아이템을 삽입한다. -1이 큐에 없으면 일반 큐처럼 동작한다.**

- **Enqueue 또는 MinDequeue가 실행되면, minimum_pos 값도 그에 맞게 수정한다.**