2020105695 김희성

실습 내용1

문제

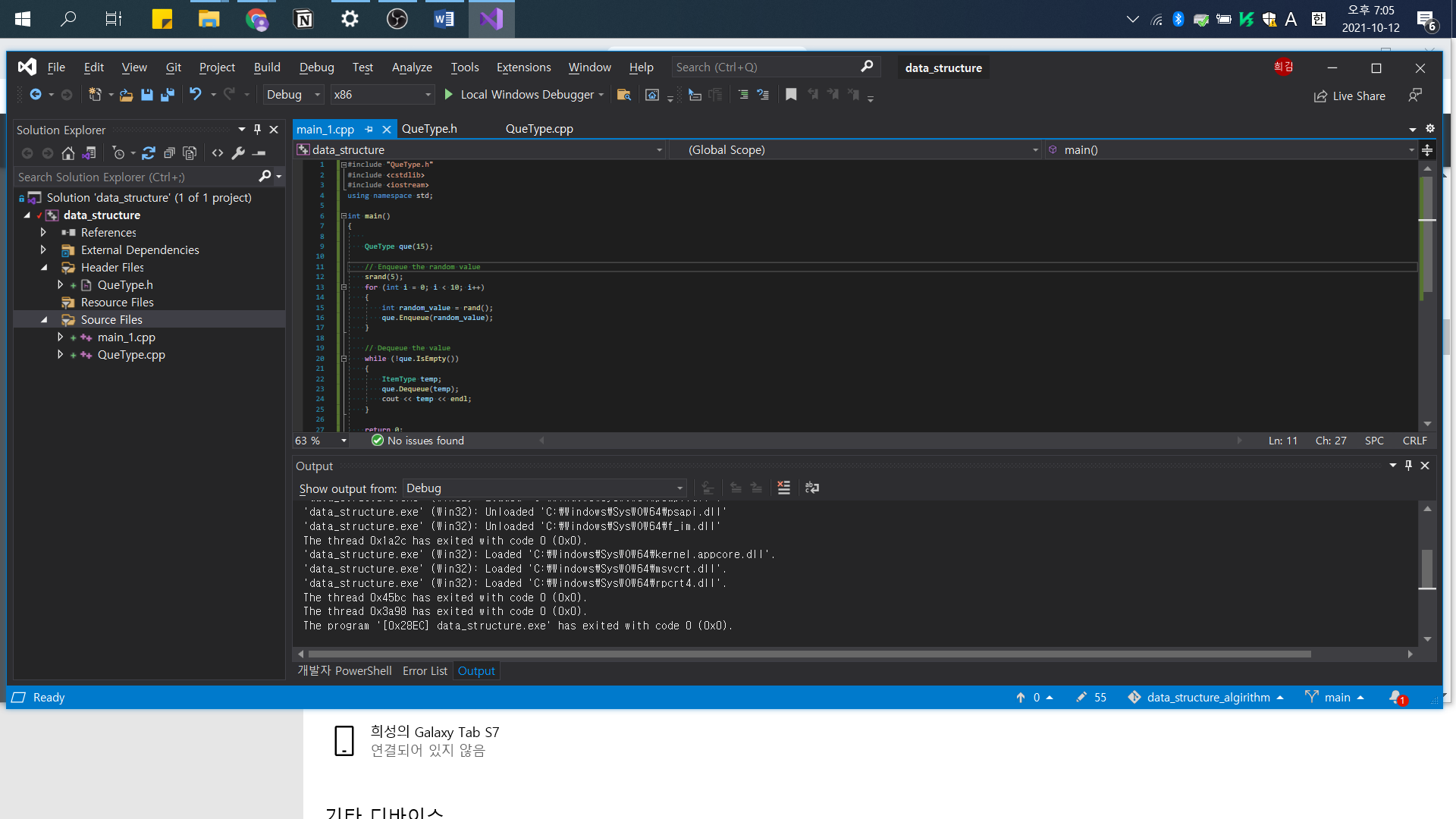
❖ 큐의 동작 방법을(Enqueue, Dequeue) 익힌다.

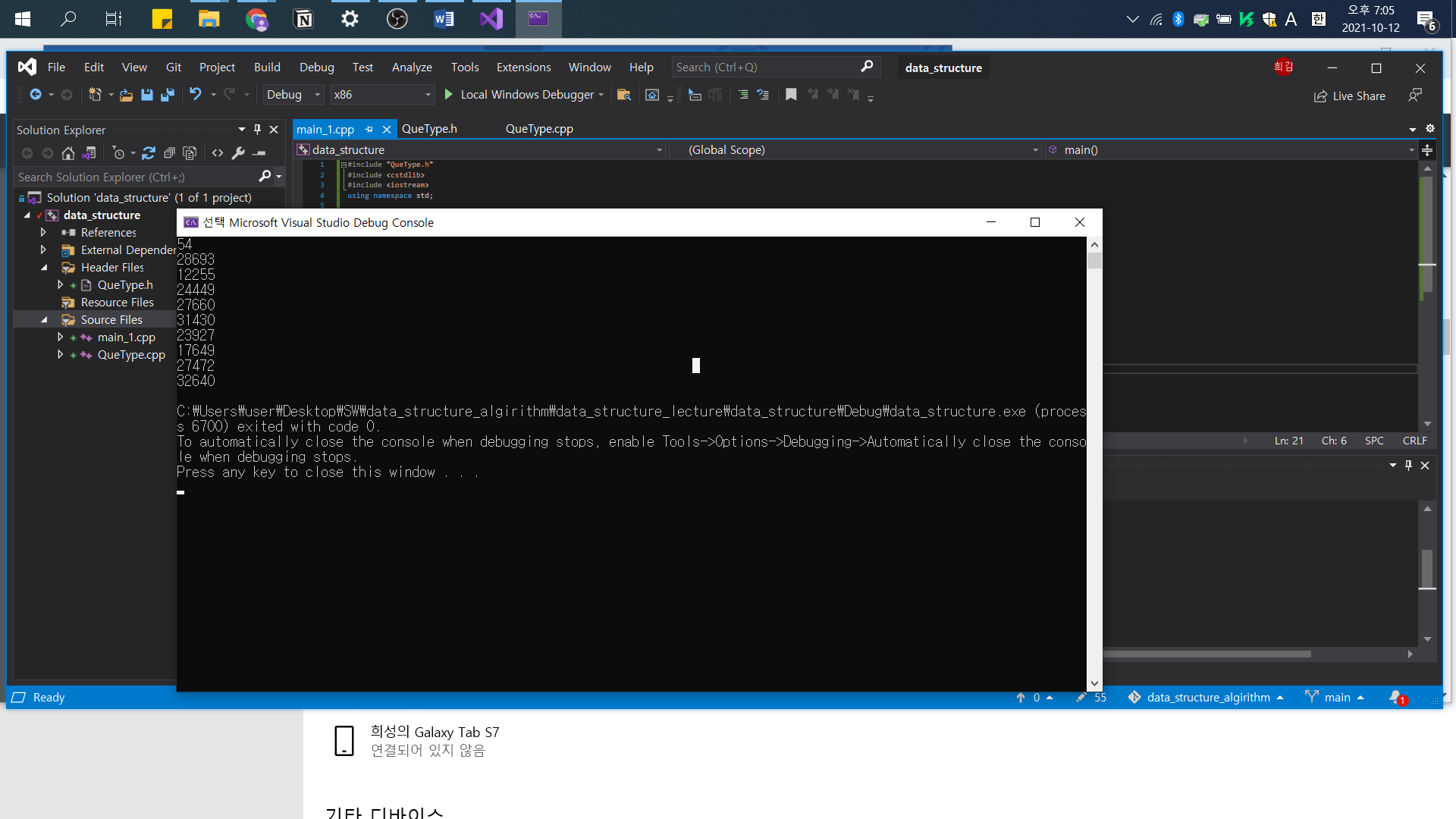
❖ 10개의 아이템을 랜덤한 값을 가지도록 생성

❖ Enqueue를 통해 아이템의 값을 저장한다. 큐에 아이템을 넣기 전에 아이템 값을 출력

❖ Dequeue를 통해 값을 가져와 출력한다.

//main\_1.cpp





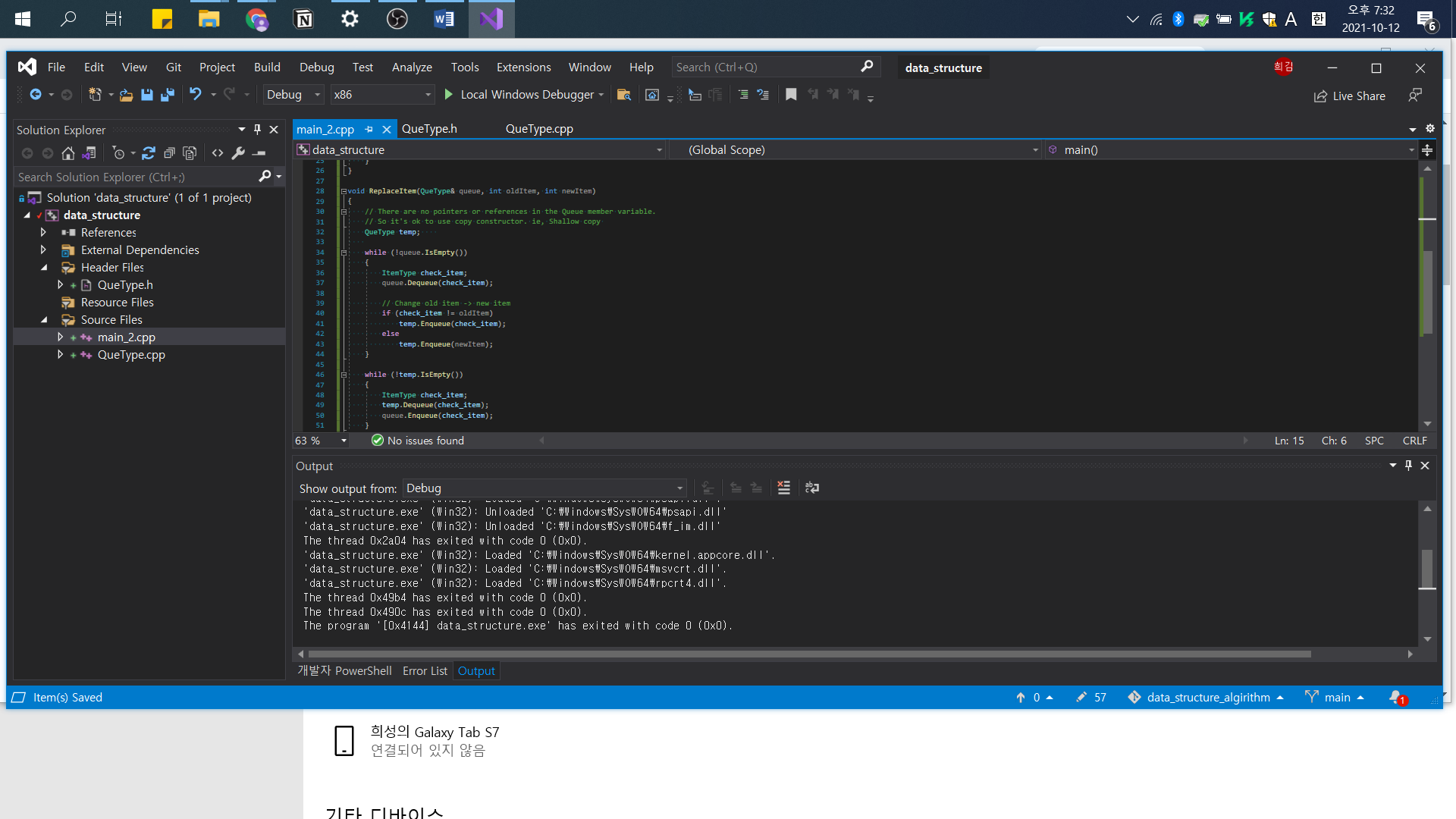
실습 내용2

ReplaceItem이란 함수를 구현하라.

❖ ReplaceItem은 내부에 존재하는 값을 새로운 값으로 치환함

❖ A. Client 함수로 작성

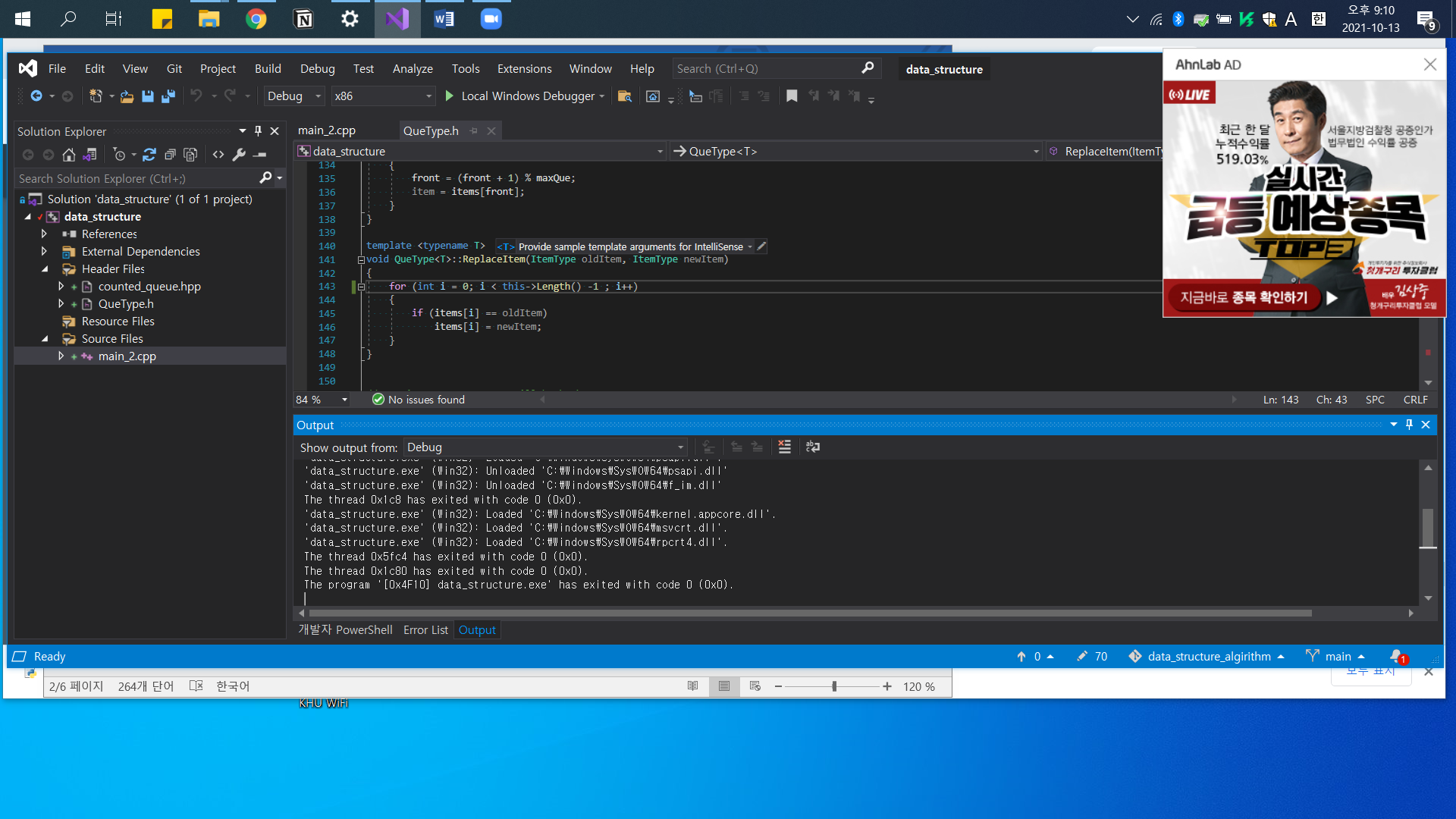
• 프로토타입 : void ReplaceItem(QueType &queue, int oldItem, int newItem);



❖ b. 멤버함수로 작성

• 프로토타입 : void ReplaceItem(ItemType oldItem, ItemType newItem);

// !! Caution !! 매우 낮은 가능성으로 잘못된 메모리 접근이 가능함.

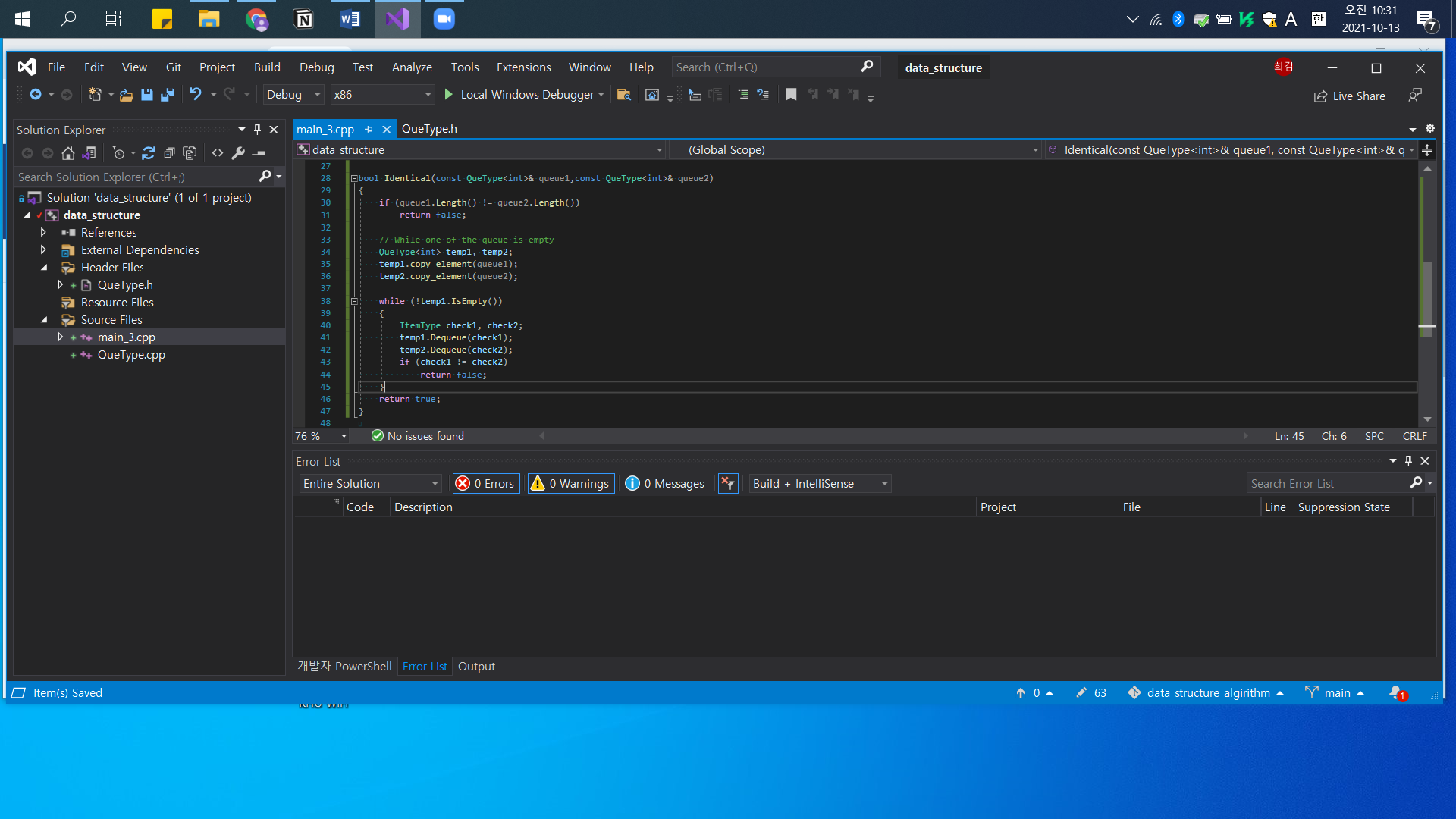


실습 내용3

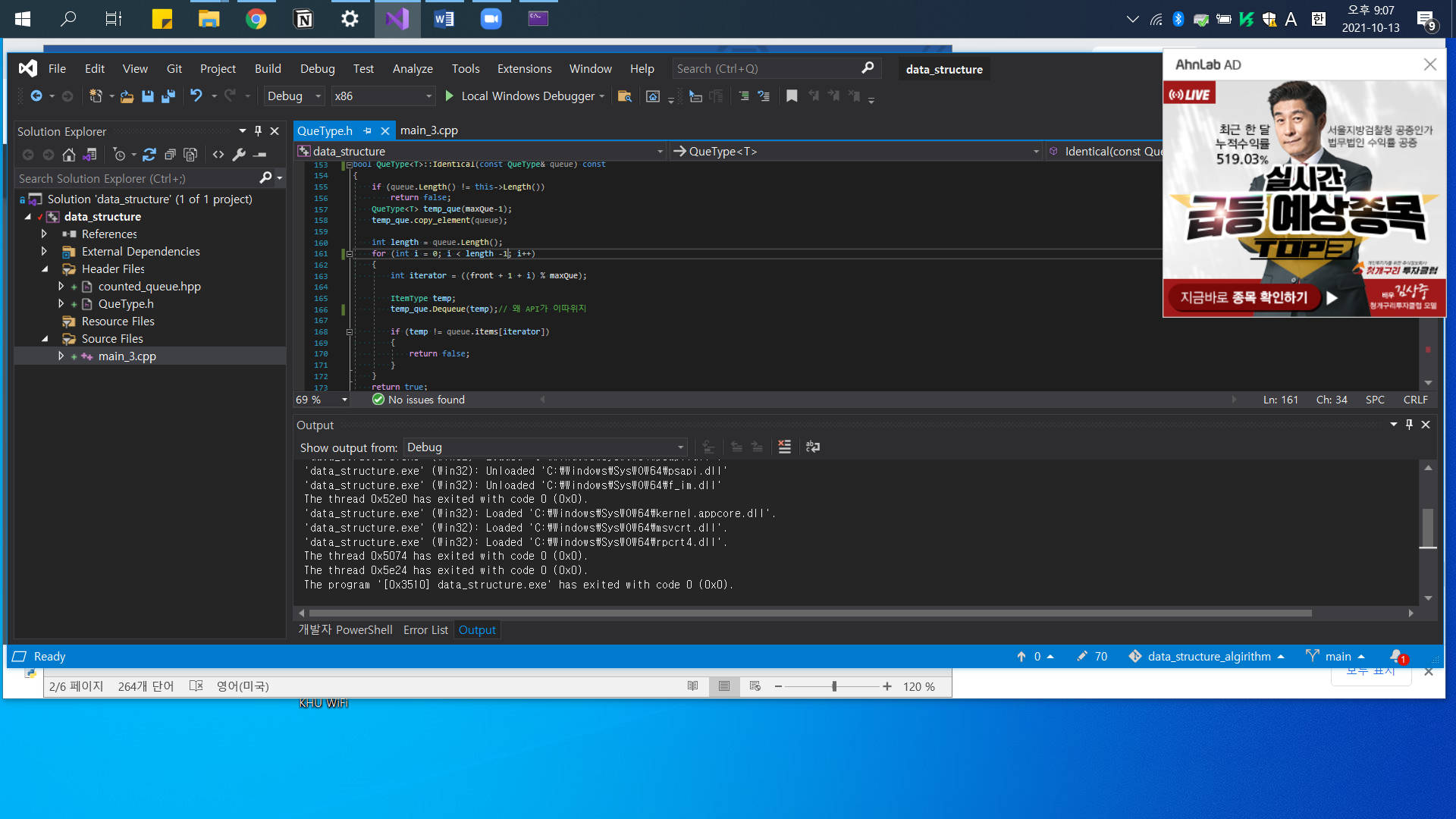
문 제

❖ 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(QueType<ItemType> queue);

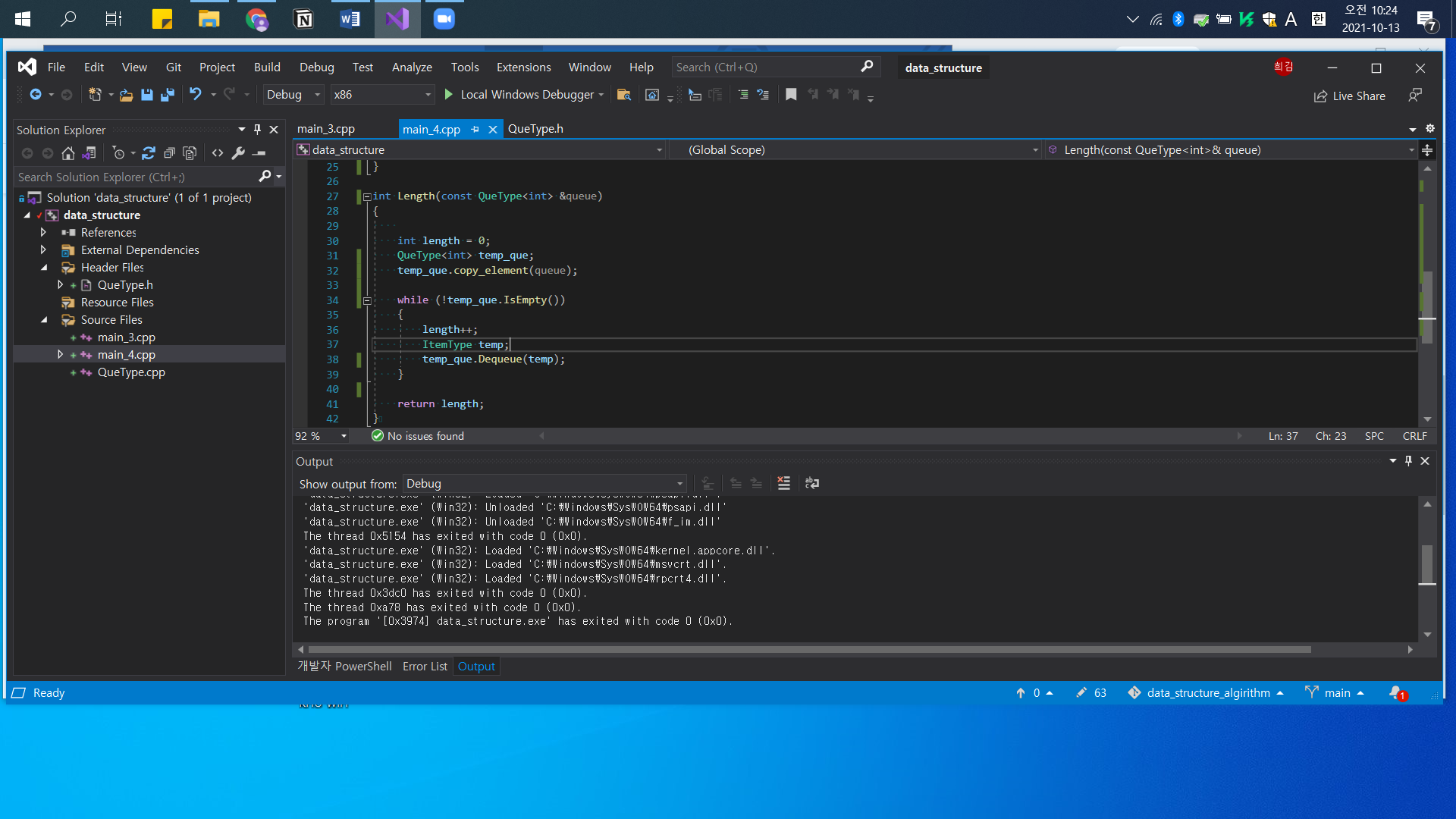


실습 내용4

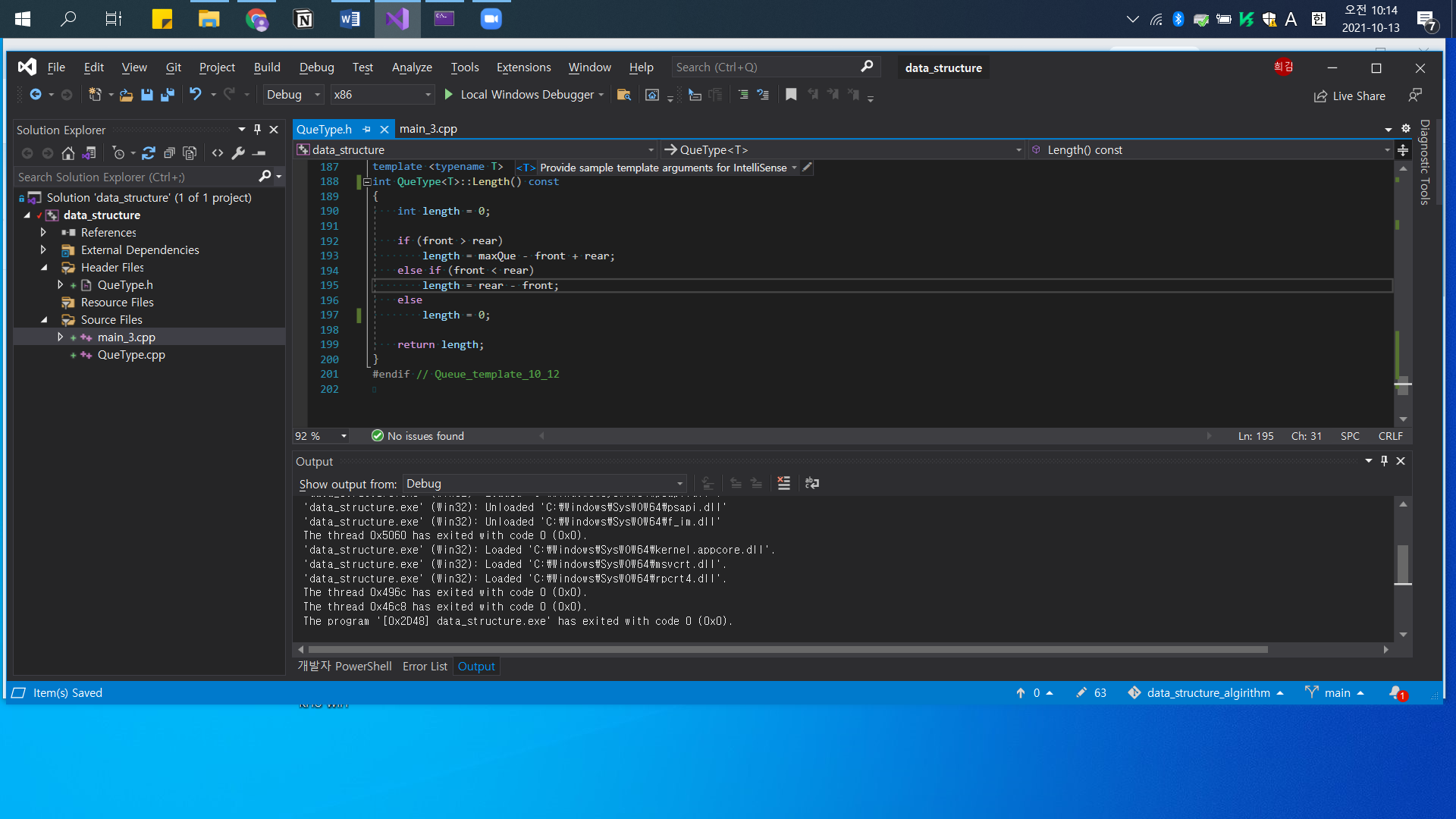
문 제

❖ 큐에 몇 개의 아이템이 저장되었는지를 리턴하는 int형의 함수를 작성하라

• a. Client 함수로 작성 - 프로토타입 : int Length(QueType<ItemType> queue);



• b. 멤버함수로 작성 - 프로토타입 : int Length();



실습 내용5

문 제

❖ 현재 구현된 큐 자료구조는 가득 찬 경우와 비어있는 경우를 구분하기 위하여 Front 앞을 비워두고 예약 공간으로 사용하였다.

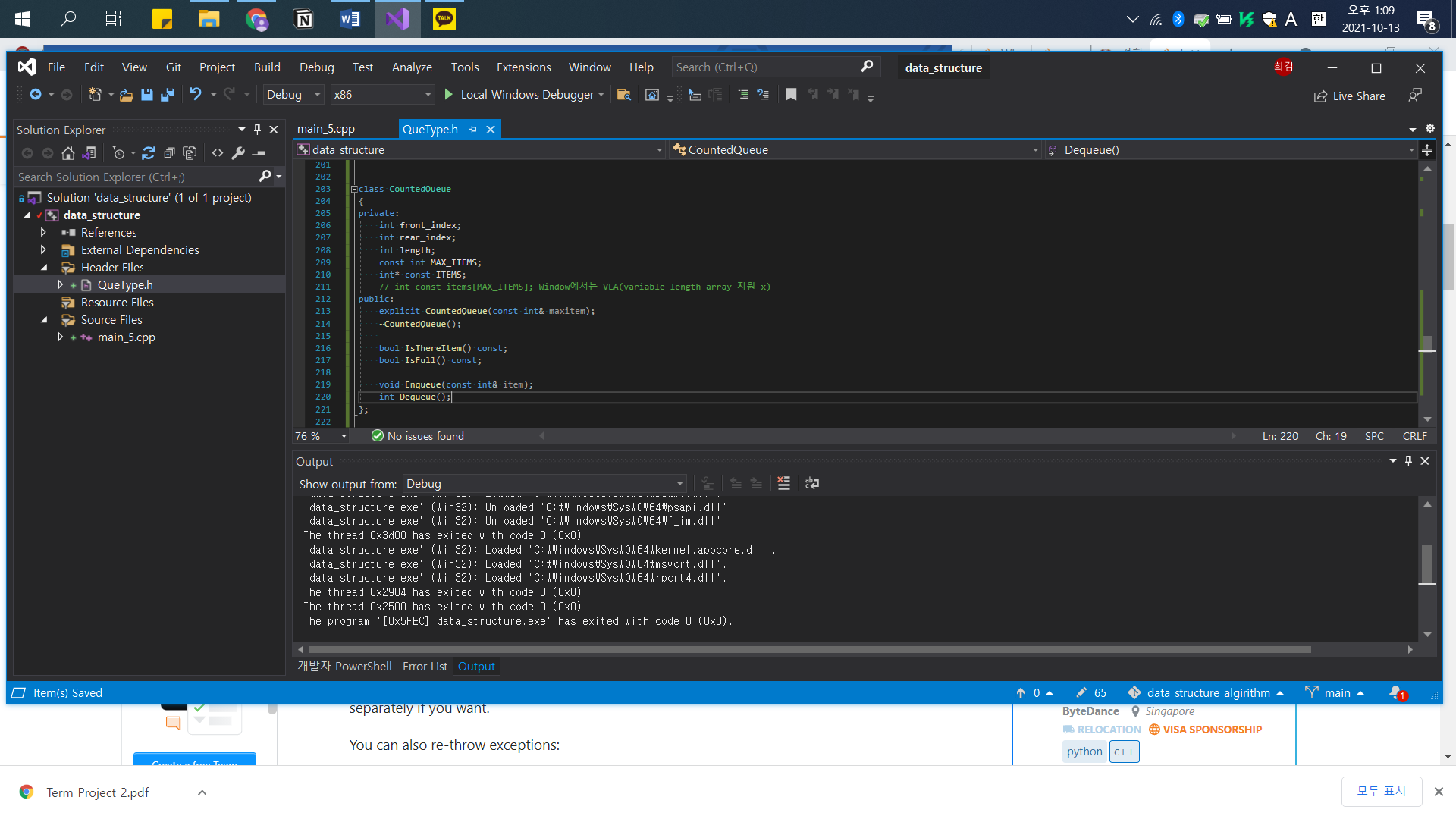
❖ 기존 큐가 Full 상태인 경우

❖ 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 Dequeue 동작을 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.

❖ MinDequeue 함수를 선언하고 수정된 내용을 이 함수에 작성한다.

O(N^2)복잡도

