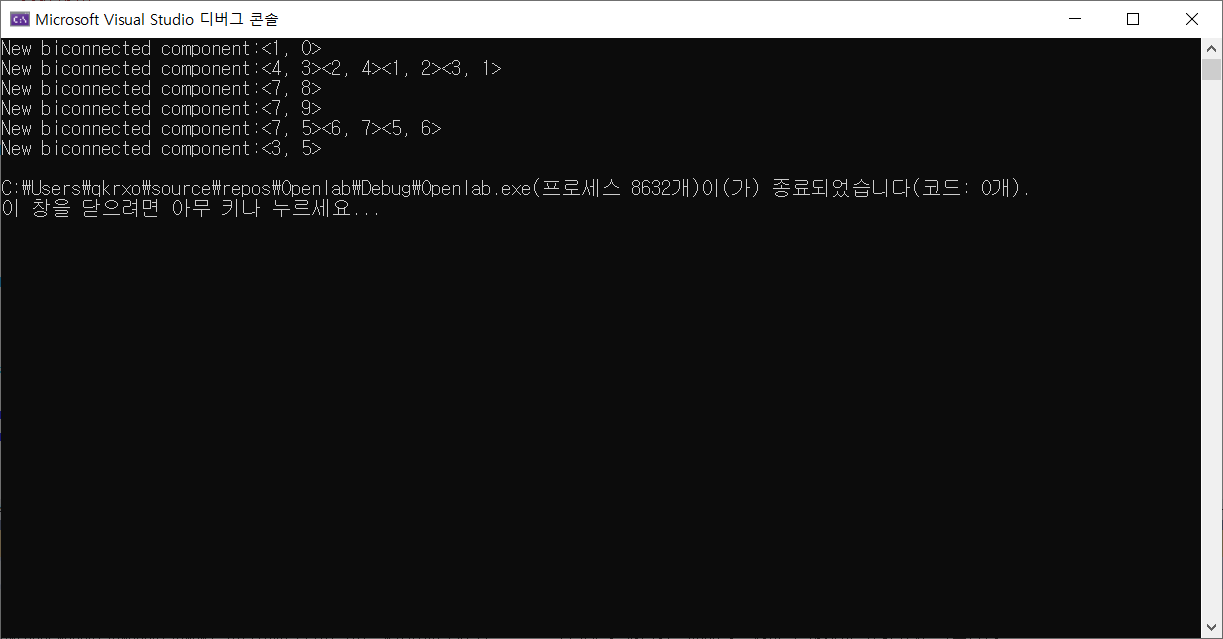
자료구조 오픈랩7

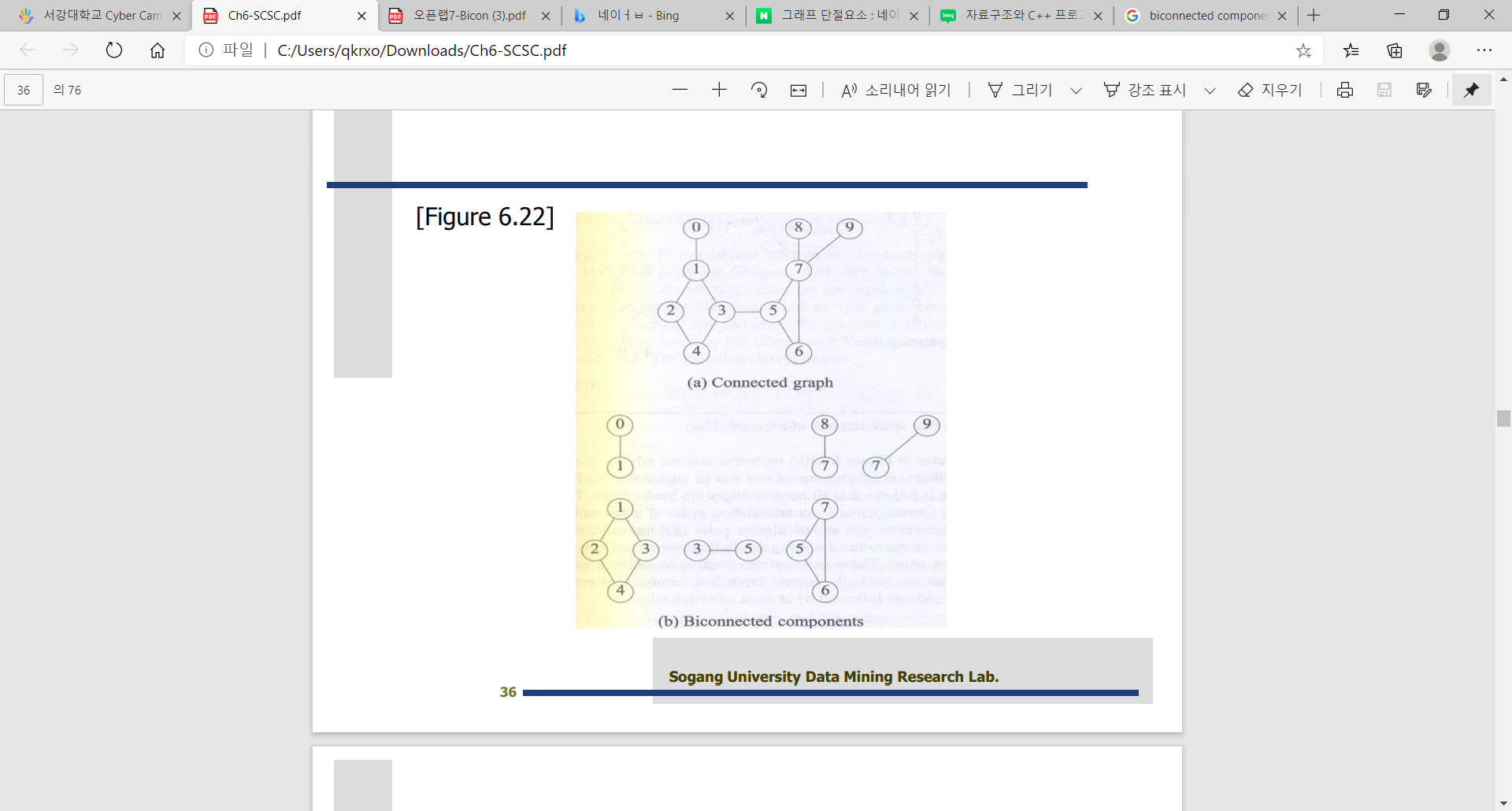
컴퓨터공학과 20171646 박태윤

**- 코드 실행 결과**



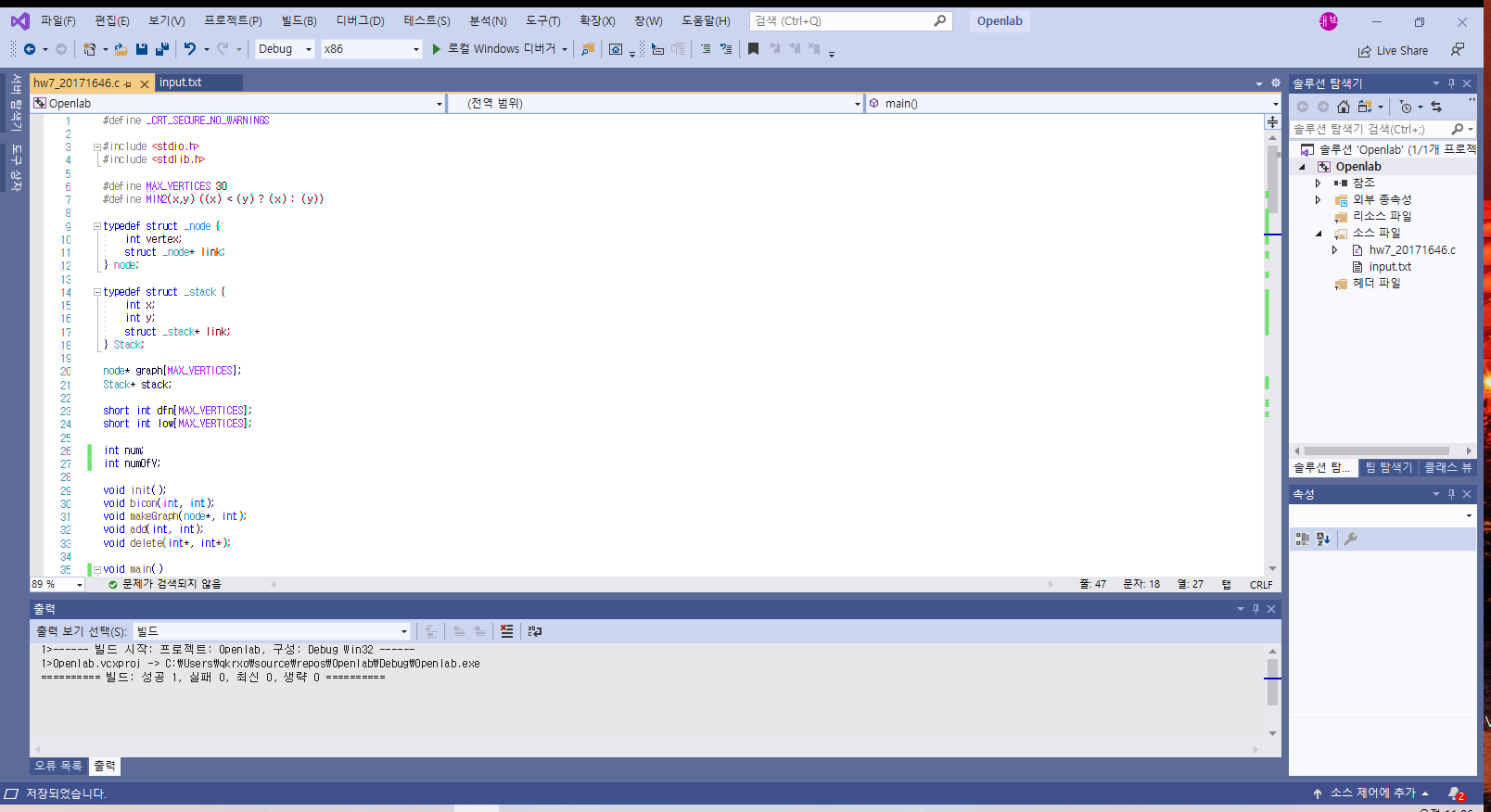
그래프의 biconnected component를 구해서 출력해주는 코드를 실행시킨 결과이다.

**- 코드 및 알고리즘 설명**

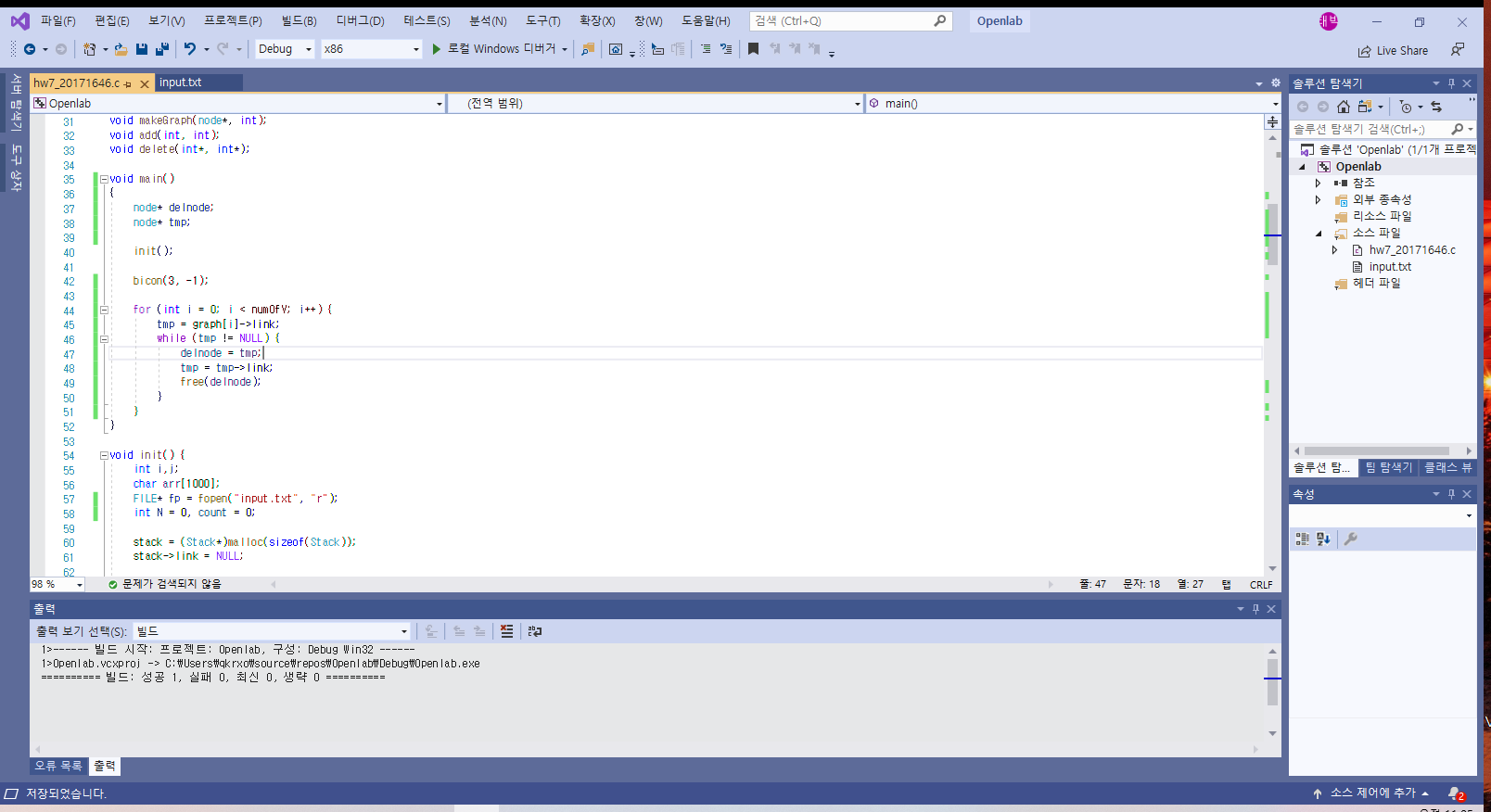


그래프에서 표시된 부분을 단절점이라고 한다. 단절점은 이 점을 기준으로 그래프를 나누었을 시 정점을 2개 이상 가지는 sub graph가 최소 2개 이상 나오게 하는 점을 의미한다. 예를 들어 위의 그래프에서 9같은 경우 9를 삭제한다고 해서 정점을 2개 이상 가지는 sub graph가 2개 이상 나오지는 않는다. 하지만 7을 삭제했을 시 8과 9 그리고 나머지 0~6으로 이루어진 그래프가 나오기 때문에 7을 단절점이라고 할 수 있는 것이다.

biconnected components는 단절점들을 기준으로 나누었을 시 나오는 부분 그래프들을 의미한다. biconndected components는 dfn과 low를 통해 나눌 수 있는데, dfn은 dfs를 통해 순서대로 매겨진 번호를 의미하고 low는 dfs를 통해 트리를 만들었을 때, 해당 정점에서 도달 가능한 가장 낮은 dfn을 의미한다.



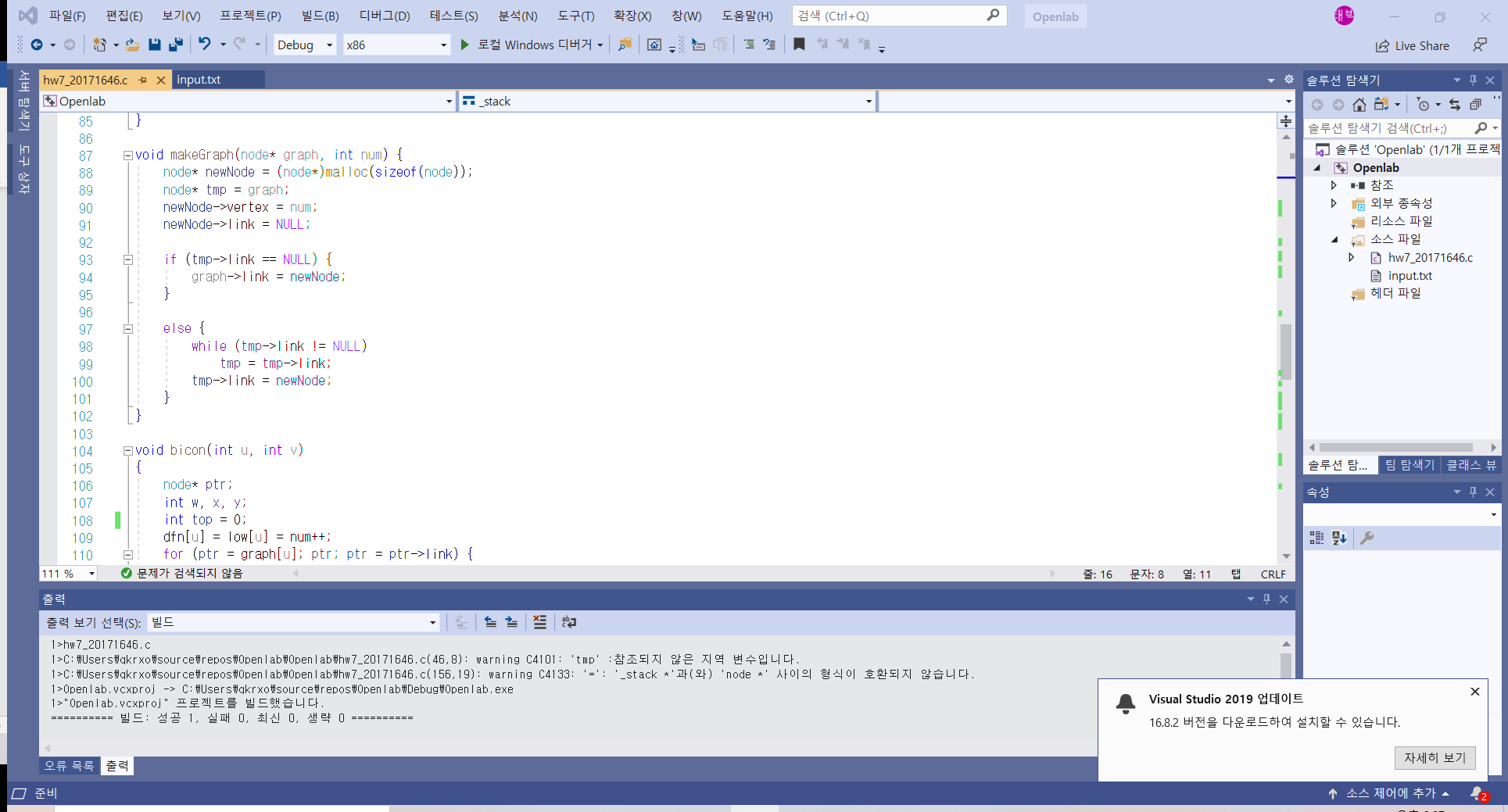
그래프를 표현하기 위한 요소 node와 연결리스트로 스택을 구현하기 위해 Stack을 선언하였다. 그래프를 표현하는 node\*배열 graph[]을 선언하였으며 앞에서 언급한 dfn과 low를 표현하는 dfn[], low[]을 선언하였다. num은 bicon함수에서 dfn을 표현하기 위한 전역변수이다. Init함수는 연결리스트 배열 graph에 그래프의 정보를 만들어서 저장해주며 stack, num, dfn, low를 초기화해준다. bicon은 biconnected components를 구한 뒤 출력해준다. makeGraph함수는 Init함수에서 사용되는데 그래프 연결리스트에 새로운 노드를 연결해주는 역할을 한다. 나머지 add와 delete는 스택에서 push와 pop같은 역할을 한다.



메인함수는 다음과 같다. 처음에 init함수를 호출한 뒤 bicon(3, -1)를 호출하는데, 이는 그래프에서 vertex가 3인 노드부터 시작한다는 의미를 가지고 있다. 함수들을 모두 실행시키고 나서 메모리 해제를 한다.

|  |  |
| --- | --- |
|  |  |

init함수에서 input.txt를 읽어들인다. 또한 전역변수로 선언된 stack또한 초기화를 시켜준다. 이후 텍스트파일 첫줄에 정점의 개수가 주어지기 때문에 fscanf를 통해 numOfV변수에 정점의 개수를 assign해준다. 정점의 개수를 받았으므로 graph배열 각 인덱스의 첫 노드를 초기화 시켜준다. 그 다음 fgets함수를 한 번 호출하여 텍스트 파일에서 정점의 개수가 쓰여진 줄의 엔터(‘\n’)를 받아주고 for반복문을 실행한다. 반복문 처음에 fgets으로 한 줄을 읽어들인 뒤 또 for반복문을 실행하는데 이 반복문은 fgets을 통해 값을 할당한 arr배열의 arr[j]가 ‘\n’나 ‘\0’을 나타낼 시 break를 하는데, 이는 즉 한 줄을 모두 읽었거나 텍스트 파일을 다 읽었을 시 반복문을 종료하는 것을 의미한다. ‘\n’나 ‘\0’이 아니라면, arr[j]가 숫자를 나타내며 해당 줄에서 처음 나오는 숫자가 아니라면(count가 0이 아님) makeGraph함수를 호출하여 graph[i]에 새로운 노드를 연결시킨다. 이후 반복문이 모두 종료되면 dfn과 low배열을 모두 -1로 초기화 시켜주고 num=0으로 초기화시킨다.

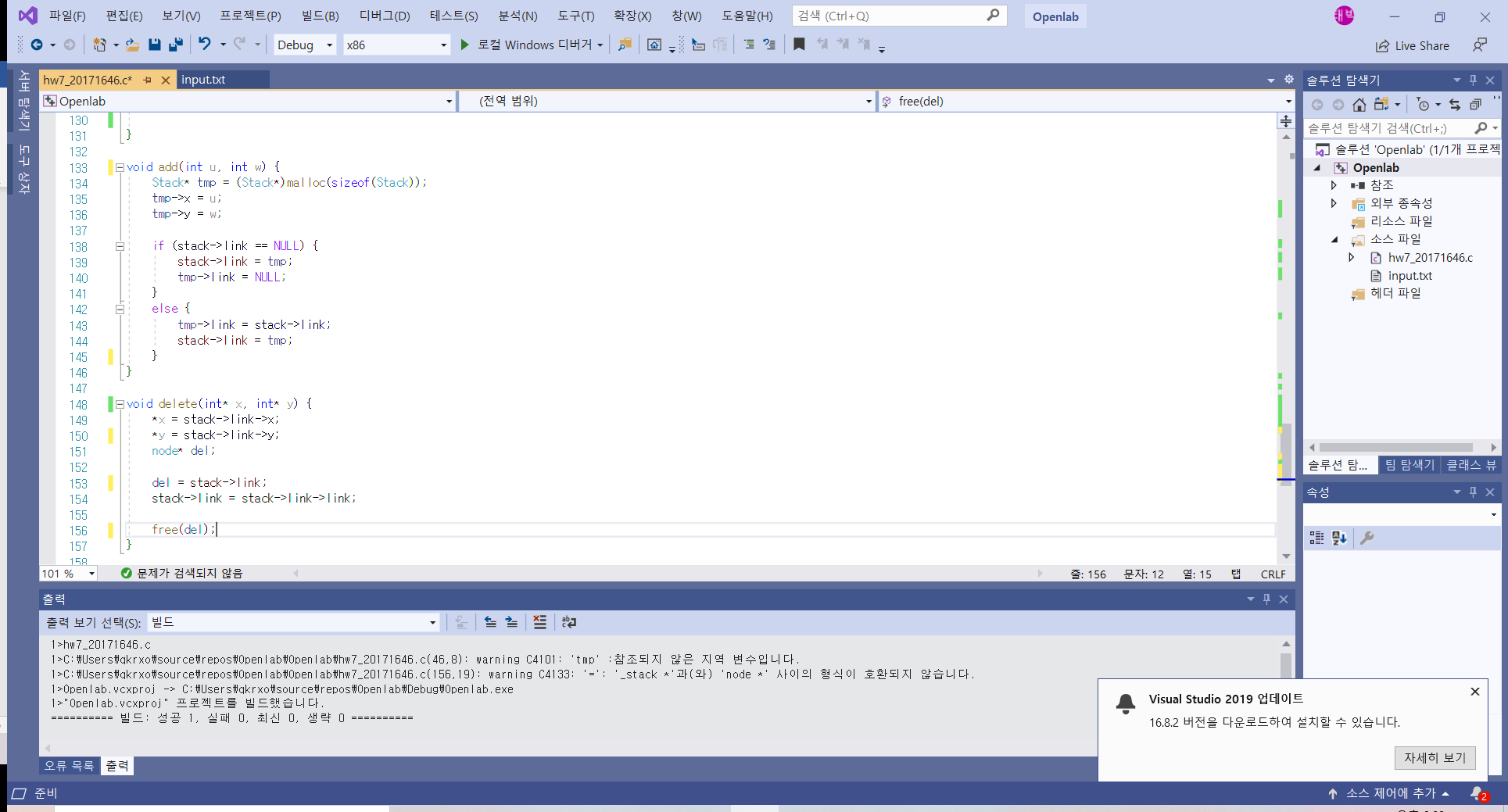


새로운 노드를 만들어 graph에 연결을 해주는 함수이다. 연결은 텍스트 파일에 쓰여진 순서대로 연결이 되는데, 이는 즉 graph[i] -> 노드1 -> 노드2 -> …. -> 노드n -> (새로운 노드) 순을 연결되는 것을 의미한다. graph에 해당 정점 이외에 아무런 노드가 연결이 되어있지 않은 경우와 그렇지 않은 경우를 나누어서 새로운 노드인 newNode를 연결시켰다.

텍스트이(가) 표시된 사진

자동 생성된 설명

bicon함수에서 dfn[u]와 low[u]를 num++로 assign해준다. num은 dfs순서를 의미하는데 처음에 bicon 호출을 bicon(3, -1)로 하였으므로 dfn[3] = 0이 된다. 이후 for반복문을 실행하는데, 반복문의 범위는 특정 정점에서부터 그 정점에 연결리스트로 연결된 마지막 정점까지로 볼 수 있다. 예를 들어 텍스트파일에 1번 정점은 1 0 2 3으로 적혀져 있는데, 반복문에서 u=1이라면 ptr=graph[1]부터 시작하여 1->0->2->3을 보는 것이다. 스택에는 간선을 추가해준다. 간선은 자식의 dfn이 부모의 dfn보다 작을 때 추가해준다. init함수에서 dfn을 -1로 초기화 하였는데 dfn이 -1이라면 dfs에서 아직 방문하지 않는 노드를 뜻한다. 때문에 표시한 부분의 if조건문에 걸리게 된다. low(u)는 dfn(u), u의 자식노드 중 가장 작은 low값, back edge로 연결된 w의 dfn중 가장 작은 값 3개 중에 가장 작은 값으로 결정된다. 즉, low(u) = min {dfn(u), min{low(w) | w is a child of u}, min{dfn(w) | (u,w) is a back edge}}으로 볼 수 있다. 만약 자식노드의 low가 부모노드의 dfn보다 크거나 같은 순간이 생긴다면, 해당 자식노드와 부모노드에 도달할 때까지 스택에 있는 간선들을 delete시켜주고 출력을 해준다. 이는 자식노드가 부모의 조상노드에 도달하지 못하는 노드일 때를 의미한다. 이 때 부모노드가 단절점이 되고 이에 대한 정보를 출력해준다.



연결리스트로 구현한 스택에서 쓰이는 add와 delete이다. delete는 노드를 삭제하면서 매개변수로 받은 x와 y에 삭제한 노드의 정보를 담기 위해 포인터형 변수로 선언하였다.