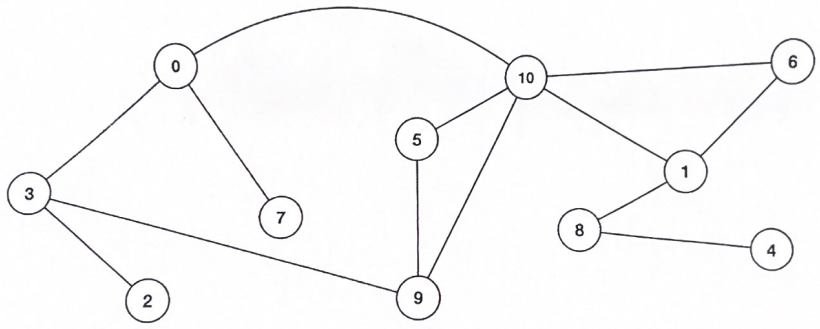
[프로그래밍 문제 답안 제출방법]

1. 하나의 폴더(folder)를 만들고 모든 소스코드를 폴더 내에 저장한다. 폴더의 이름은 "홍길동20131234"와 같이 자신의 한글 이름과 학번으로 구성한다.
2. 문제 당 하나의 소스코드 파일을 1에서 만든 폴더에 저장한다. 소스파일의 이름은 `prob*.{c, cpp, java, py}`이다. 여기서 *은 문제번호이다. 예를 들어 `prob1.c, prob2.cpp, prob3.java` 등이다. 이 파일들을 제외한 어떤 파일도 저장해서는 안된다.
3. 소스코드 폴더를 감독관이 제공하는 USB에 복사한 후 감독관의 노트북으로 복사한다. 감독관이 제공한 USB 이외의 다른 USB 메모리를 사용해서는 안된다.
4. PC에 파일들은 감독관의 노트북으로 파일 복사가 완료되었는지 확인 후 삭제한다.
5. 각 문제에 대해서 입출력 요구사항을 정확하게 준수한다. 입력의 순서와 형식을 마음대로 바꾸거나 요구된것 이외의 정보를 출력해서는 안된다.

[프로그래밍 문제]

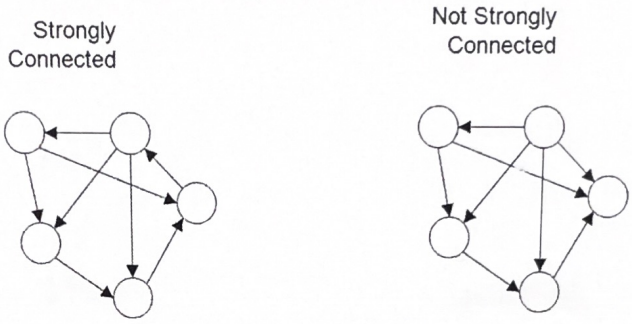
1. $N < 100$개의 정점을 가진 무방향(undirected) 그래프 G의 인접행렬이 입력으로 주어진다. 정점의 번호는 $0,1,\ldots,N-1$이다. 또한 하나의 정수 $k < 10$가 추가로 주어진다. 이 그래프의 각 정점 v에 대해서 그 정점에서 k칸 이내의 거리에 있는 정점의 개수가 최대가 되는 정점을 찾아서, 찾아진 정점의 번호와 정점의 개수를 출력하는 프로그램을 작성하라. 예를 들어 아래의 그래프에서 $k = 2$인 경우 정점 10으로부터 거리 2 이내에 있는 정점들은 `10, 0, 3, 7, 5, 9, 1, 6, 8`이며 개수는 9개로 최대이다(자기 자신을 포함하여 카운트한다.) 따라서 이 경우 출력은 `10, 9`이다.



입력 예	출력
9 // N 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 2 // k	6 7

입력 예	출력
11 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 1 1 0 0 0 1 1 0 0 1 0 2	10 9

2. 방향 그래프에서는 정점 u에서 v로 가는 경로가 존재한다고 해서 반드시 v에서 u로 가는 경로가 존재하지는 않는다. 그래서 방향 그래프의 연결성의 개념은 무방향 그래프보다는 조금 더 복잡하다. 방향 그래프 G의 **임의의 두 정점 u와 v에 대해서 u에서 v로 가는 경로가 존재할 때 G는 강연결**(strongly connected) **그래프**라고 말한다. 아래 그림은 강연결 그래프와 강연결이 아닌 그래프의 예이다.



그래프가 강연결 그래프인지 검사하는 한 가지 방법은 각각의 정점들에 대해서 그 정점에서 출발하는 DFS 혹은 BFS를 해서 각각의 정점에서 다른 모든 정점으로 갈 수 있는지 검사해 보는 것이다. 인접행렬을 사용할 경우 DFS 혹은 BFS를 N번 수행하게 되므로 시간복잡도는 $O(N^3)$이 된다. 하지만 다음과 같이 좀 더 효율적으로 검사할 수 있다.

(1) 우선 임의의 정점 s를 선택하여 s에서 출발하는 BFS 혹은 DFS를 수행하여 정점 s에서 다른 모든 정점으로 가는 경로가 있는지 검사한다.
(2) 그래프 G의 모든 에지들의 방향을 뒤집은 그래프를 G^c라고 하자.
(3) G^c에서 다시 s에서 출발하는 BFS 혹은 DFS를 수행하여 정점 s에서 다른 모든 정점으로 가는 경로가 있는지 검사한다. G^c에서 s에서 다른 모든 정점으로 가는 경로가 존재한다면 이것은 원래의 그래프 G에서는 다른 모든 정점에서 s까지 가는 경로가 존재함으로 의미한다.

이 두 번의 검사를 통과하는 것이 강연결 그래프이기 위한 필요충분조건임은 쉽게 증명할 수 있다. 이 알고리즘은 2번의 DFS 혹은 BFS를 하면 되므로 시간복잡도는 $O(N^2)$이다. 입력으로 하나의 방향 그래프를 받아서 강연결 그래프인지 검사하여 YES 혹은 NO를 출력하는 프로그램을 작성하라. 입력은 키보드로 주어지고, 입력의 첫 줄에는 정점의 개수 $N < 100$이 주어지고 이어진 N줄에는 각 줄 마다 하나의 정점 번호와 그 정점에 인접한 정점들의 개수 및 번호가 주어진다.

입력 예		출력
5 0 2 2 3 1 3 0 2 4 2 1 4 3 1 1 4 1 3	// N = 5 // 0번 정점에 인접한 정점은 2개이며 번호는 2, 3이다. // 1번 정점에 인접한 정점은 3개이며 번호는 0, 2, 4이다. // 2번 정점에 인접한 정점은 1개이며 번호는 4이다.	YES

입력 예	출력
5 0 2 2 3 1 4 0 2 3 4 2 1 4 3 0 4 1 3	NO