


[Algorithm/알고리즘](#)

## [BOJ](Python) 백준 5567번: 결혼식

그리브 | 2023. 11. 21. 11:40 | 수정 | 삭제

### 문제

문제  
풀이<https://acmicpc.net/problem/5567>



### 5567번: 결혼식

예제 1의 경우 2와 3은 상근이의 친구이다. 또, 3과 4는 친구이기 때문에, 4는 상근이의 친구의 친구이다. 5와 6은 친구도 아니고, 친구의 친구도 아니다. 따라서 2, 3, 4 3명의 친구를 결혼식에 초대

[www.acmicpc.net](http://www.acmicpc.net)

주제	그래프 구현/순회
시간 / 메모리 제한	1초 / 128MB
정답 비율	44.052%

상근이는 자신의 결혼식에 학교 동기 중 자신의 친구와 친구의 친구를 초대하기로 했다.

상근이의 동기는 모두  $N$ 명이고, 이 학생들의 학번은 모두 1부터  $N$ 까지이다.

상근이의 학번은 1이다.

상근이는 동기들의 친구 관계를 모두 조사한 리스트를 갖고 있다.

이 리스트를 바탕으로 결혼식에 초대할 사람의 수를 구하는 프로그램을 작성하라.

### 입력

첫째 줄에 상근이의 동기의 수  $n(2 \leq n \leq 500)$ 이 주어진다.

둘째 줄에는 리스트의 길이  $m(1 \leq m \leq 10000)$ 이 주어진다.

다음 줄부터  $m$ 개의 줄에는 친구 관계  $a_i, b_i(1 \leq a_i < b_i \leq n)$ 이 주어진다.

- $a_i$ 와  $b_i$ 가 친구라는 뜻이며,  $b_i$ 와  $a_i$ 도 친구관계이다.

### 출력

첫째 줄에 상근이의 결혼식에 초대하는 동기의 수를 출력한다.

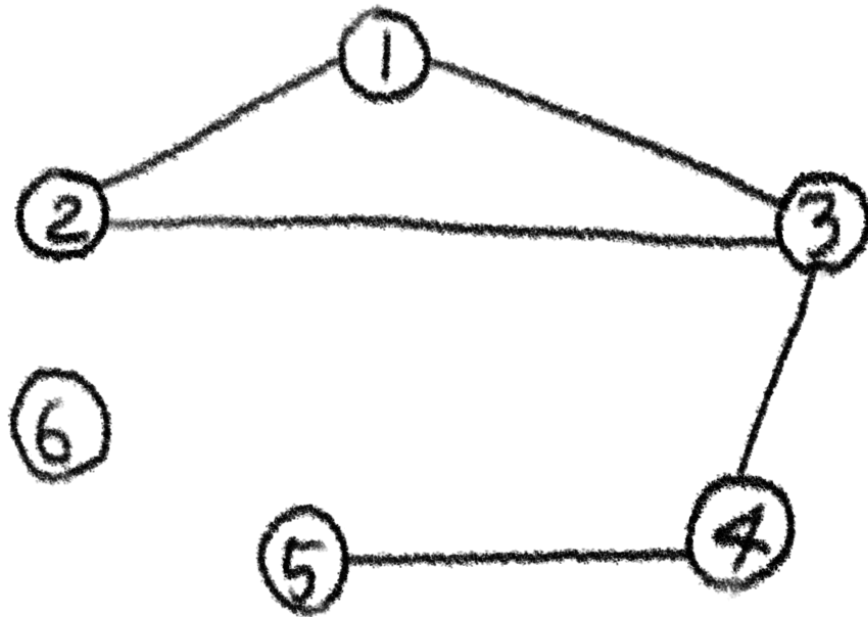
- 예제 입력 1

Copy

```
1 6
2 5
```

```
3 1 2
4 1 3
5 3 4
6 2 3
7 4 5
```

python



- 예제 출력 1

```
1 3
```

Copy

python

## 힌트

예제 1의 경우 2와 3은 상근이의 친구이다.

또, 3과 4는 친구이기 때문에, 4는 상근이의 친구의 친구이다.

5와 6은 친구도 아니고, 친구의 친구도 아니다.

따라서 2, 3, 4 3명의 친구를 결혼식에 초대한다.

- 예제 입력 2

```
1 6
2 5
3 2 3
4 3 4
5 4 5
6 5 6
7 2 5
```

Copy

python

- 예제 출력 2

```
1 0
```

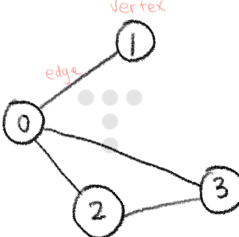
Copy

python

## 풀이

그래프에 대한 자세한 개념은 작성해둔 아래의 포스트에서 학습하면 된다.

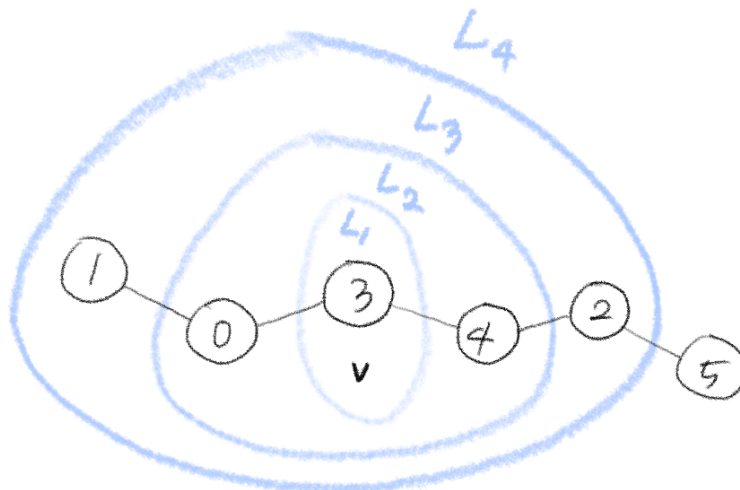
<https://classic-griver.tistory.com/192>

	<p><b>[알고리즘](Python) 그래프 개념과 파이썬 코드 ...</b></p> <p>참고 자료: 파이썬으로 배우는 자료 구조 핵심 원리(양태환) 6.1 그래프 용어 정리 그래프는 정점(vertex)과 에지(edge)로 이루어진다. vertex는 어떤 대상의 객체를 말하고 edge는 그 vertex를 ...</p> <p>classic-griver.tistory.com</p>
-----------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

필요한 메서드는 위의 링크에서 가져와 사용하였다.

이 문제는 그래프의 너비 우선 탐색 즉, BFS를 수행하는 문제이다.

### \* 너비 우선 탐색 (BFS)



⇒  $v$ 가 있는  $L_1$ 의 모든 vertex  $\rightarrow L_2$ 의 모든 vertex  $\rightarrow L_3 \dots \rightarrow L_4$

본인에 해당하는 1번 vertex를 기준으로 하여,  $L_1$ 으로 지정하고

인접 vertex로 퍼져나가  $L_2, L_3$ 에 해당하는 vertex의 총 개수를 세어주면 된다.

위 그림을 참고하여 아래에 사용한 명칭은 다음을 나타낸다.

- $L_1$ : 상근이
- $L_2$ : 상근이의 친구
  - $L_1$ 의 인접리스트
- $L_3$ : 상근이의 친구의 친구
  - $L_2$ 의 인접리스트

문제를 풀며 BOJ에서 '틀렸습니다.'라는 결과를 두 번 마주하였는데 그 풀이 중 하나가 아래의 풀이다.

## 오답 1 - L1의 인접 리스트만 순회, L2의 인접 리스트 비순회

```
1 import sys
2
3 class Graph:
4
5     def __init__(self, vertex_num=None):
6         self.adj_list = []
7         self.vtx_num = 0
8         self.vtx_arr = []
9         if vertex_num:
10             self.vtx_num = vertex_num
11             self.vtx_arr = [True for _ in range(self.vtx_num)]
12             self.adj_list = [[] for _ in range(self.vtx_num)]
13
14     def add_edge(self, u, v):
15         self.adj_list[u].append(v)
16         self.adj_list[v].append(u)
17
18 n = int(sys.stdin.readline().strip())
19 m = int(sys.stdin.readline().strip())
20
21 g = Graph(n)
22 for _ in range(m):
23     l = list(map(int, sys.stdin.readline().strip().split()))
24     g.add_edge(l[0] - 1, l[1] - 1)
25
26 res = 0
27 for i in g.adj_list[0]:
28     res += len(g.adj_list[i]) - 1
29
30 print(res)
```

Copy

python

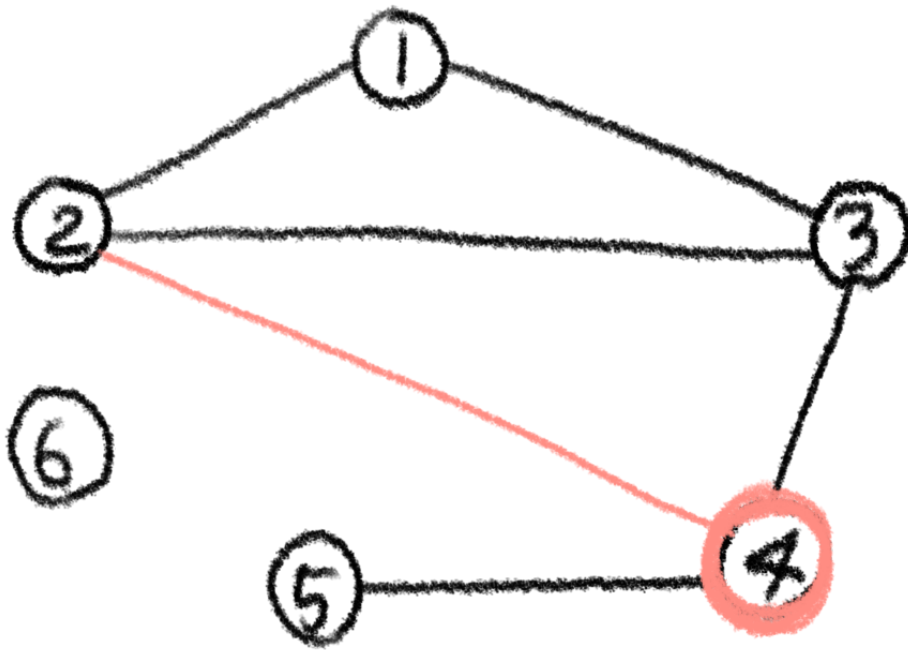
주의해야 할 점은, 코드 상 vertex의 인덱스는 실제 친구의 이름에다가 1을 빼주어야 한다.

- 친구 2에 해당하는 vertex는 1

첫 번째 오답은 초대 인원을 세는 방법으로 L1에 해당하는 상근이(vertex 0)의 인접리스트만 직접 순회하였다.

그리고 L2에 해당하는 상근이 친구의 인접리스트는 직접 순회하지 않고 길이만 가져와 사용하였다.

이 방법의 문제점은 아래와 같은 상황에서 발생한다.



```

1  ### 그래프 연결 상태
2  num of vertices : 6
3  vertices : {0, 1, 2, 3, 4, 5, }
4  [0] : {1, 2, }
5  [1] : {0, 2, 3, }
6  [2] : {0, 3, 1, }
7  [3] : {2, 4, 1, }
8  [4] : {3, }
9  [5] : {}

```

만약 상근이의 친구의 친구가 겹친다면 이 겹치는 사람을 하나로 세지 못 하고 두 번 세게 된다는 것이다.

4번 친구의 경우 상근이의 친구인 2, 3번과 모두 친구인데

위 코드로 세게 될 경우 4번 친구가 2번이나 세어지게 된다.

따라서 단순히 상근이의 인접리스트인 L1만 순회하는 것이 아니라,

L1의 인접리스트인 L2까지 직접 순회하여

초대 리스트에 들어간 사람 간 겹치는지를 일일이 체크해주어야 된다는 것이다.

이러한 이유로 초대 리스트에 해당하는 guest 리스트를 추가하고

for 반복문을 추가하여 L1과 L2의 인접리스트를 직접 방문하는 코드로 수정하였다.

## |    오답 2 - L1, L2의 인접 리스트 순회

```

1  import sys
2
3  class Graph:
4
5      def __init__(self, vertex_num=None):

```

```

6     self.adj_list = []
7     self.vtx_num = 0
8     self.vtx_arr = []
9     if vertex_num:
10        self.vtx_num = vertex_num
11        self.vtx_arr = [True for _ in range(self.vtx_num)]
12        self.adj_list = [[] for _ in range(self.vtx_num)]
13
14    def add_edge(self, u, v):
15        self.adj_list[u].append(v)
16        self.adj_list[v].append(u)
17
18    n = int(sys.stdin.readline().strip())
19    m = int(sys.stdin.readline().strip())
20
21    g = Graph(n)
22    for _ in range(m):
23        l = list(map(int, sys.stdin.readline().strip().split()))
24        g.add_edge(l[0] - 1, l[1] - 1)
25
26    guest = [0]
27    for i in g.adj_list[0]:
28        for j in g.adj_list[i]:
29            if j not in guest:
30                guest.append(j)
31
32    print(len(guest) - 1)

```

python

L1, L2의 인접리스트를 순회하며 guest 리스트에 초대할 친구들을 추가하는 방식으로 코드를 작성하였다.

재귀를 수행할 수도 있지만 반복문을 사용하는 것이 더 쉽고 직관적이라 반복문을 사용했다.

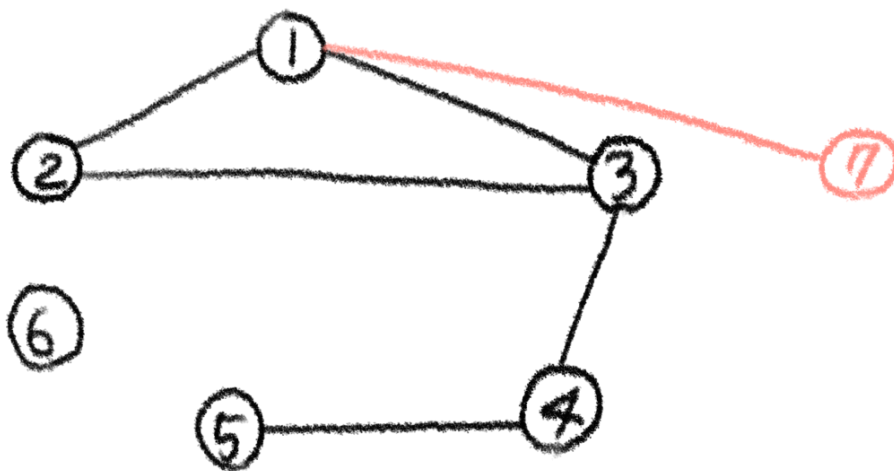
(시간 초과 시 재귀를 하려 했지만 문제가 없어 수정하지 않았다.)

하지만 위와 같은 코드에도 '틀렸습니다'라는 답변을 받았다.

## 정답 - 발생하는 예외 수정

예외를 찾을 수 있었는데, 위 코드 같은 경우에는 상근이의 친구가 또 다른 친구를 가지지 않는 경우에는 세어주지 않았다.

즉, 아래의 친구 7과 같이 L2만 있고 L3가 없는 경우에 L2의 수를 세어주지 않았다.



이를 수정한 코드를 아래와 같이 작성하였다.



```

1  import sys
2
3  class Graph:
4
5      def __init__(self, vertex_num=None):
6          self.adj_list = []
7          self.vtx_num = 0
8          self.vtx_arr = []
9          if vertex_num:
10             self.vtx_num = vertex_num
11             self.vtx_arr = [True for _ in range(self.vtx_num)]
12             self.adj_list = [[] for _ in range(self.vtx_num)]
13
14      def add_edge(self, u, v):
15          self.adj_list[u].append(v)
16          self.adj_list[v].append(u)
17
18
19  n = int(sys.stdin.readline().strip())
20  m = int(sys.stdin.readline().strip())
21
22  g = Graph(n)
23  for _ in range(m):
24      l = list(map(int, sys.stdin.readline().strip().split()))
25      g.add_edge(l[0] - 1, l[1] - 1)
26
27  guest = [0]
28  for i in g.adj_list[0]:
29      if i not in guest:
30          guest.append(i)
31      for j in g.adj_list[i]:
32          if j not in guest:
33              guest.append(j)
34
35  print(len(guest) - 1)

```

python

```

1  num of vertices : 7
2  vertices : {0, 1, 2, 3, 4, 5, 6, }
3  [0] : {1, 2, 6, }
4  [1] : {0, 2, }
5  [2] : {0, 3, 1, }
6  [3] : {2, 4, }
7  [4] : {3, }
8  [5] : {}
9  [6] : {0, }
10
11 guest: [0, 1, 2, 3, 6]

```

Copy

python

기존에는 2중 for문에서만 친구의 인접리스트를 guest에 추가해주었지만

1중 for문에서도 상근이의 인접리스트를 guest에 추가하도록 하여 발생했던 누락을 수정했다.

위 코드로 제출하여 아래와 같이 정답을 얻을 수 있었다.

제출 번호	아이디	문제	결과	메모리	시간	언어	코드 길이	제출한 시간
69514409	aprilwldnjs	5567	맞았습니다!!	31120 KB	52 ms	Python 3	822 B	44분 전

공감

Algorithm > 알고리즘 카테고리의 다른 글

[BOJ](Python) 백준 1920번: 수 찾기 (1)	2023.11.14
[BOJ](Python) 백준 1620번: 나는야 포켓몬 마스터 이다솜 (0)	2023.11.07
[휴학 중 코테 부수기] # 5. 정렬 알고리즘 (0)	2023.11.06
[휴학 중 코테 부수기] # 4. 그래프 탐색 알고리즘: DFS/BFS (2)	2023.11.06
[휴학 중 코테 부수기] # 3. 구현(Implementation) (1)	2023.11.04

'Algorithm/알고리즘' Related Articles

메모리	시간	메모리	
48200 KB	504 r	57692 KB	
[BOJ](Python) 백준 1920번: 수 찾기		[BOJ](Python) 백준 1620번: 나는야 포켓몬 마스터 이...	
		[휴학 중 코테 부수기] # 5. 정렬 알고리즘	
		[휴학 중 코테 부수기] # 4. 그래프 탐색 알고리즘:...	

☐ Secret

안녕하세요! 어떤 댓글이든 환영합니다! 💖

댓글달기



