

2020년 ALPS 여름방학 비대면 스터디

- 6주차 -
문제풀이

참여방법

- 슬랙 - 코드 공유용으로 사용
- 구글 meet - 화면공유 및 음성으로 코드 설명 및 서로간 피드백 (<https://meet.google.com/gpu-mixd-bqa>) 링크는 계속 고정입니다.

항상 위 링크로 들어오시면 됩니다.

슬랙과 meet를 같이 띄우고 진행해주시면 감사하겠습니다.

참여방법 / 슬랙



Join your team on Slack

has invited you to use Slack with them,
in a workspace called **ALPS Study 2020.07**.



ALPS Study 2020.07

alps-study-202007.slack.com

Join Now

has already joined



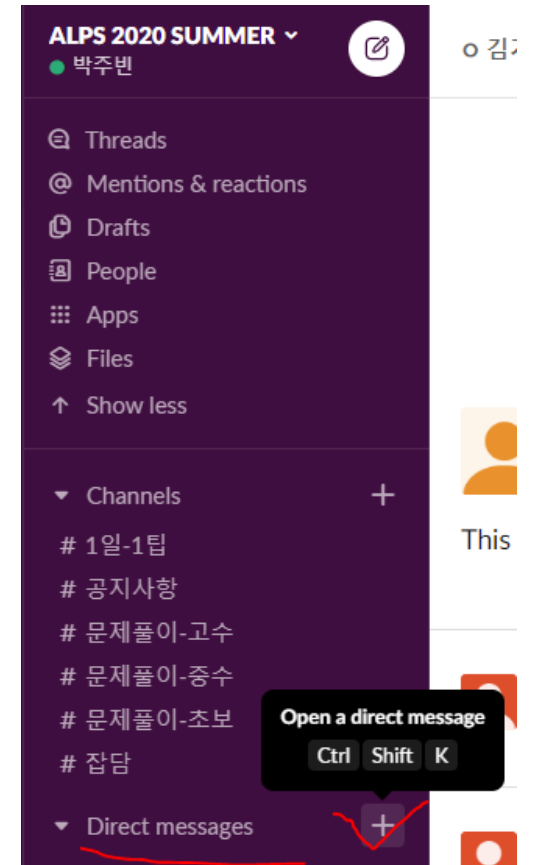
메일로 보내드린
초대장에 Join Now를
클릭하시면
입장 가능합니다

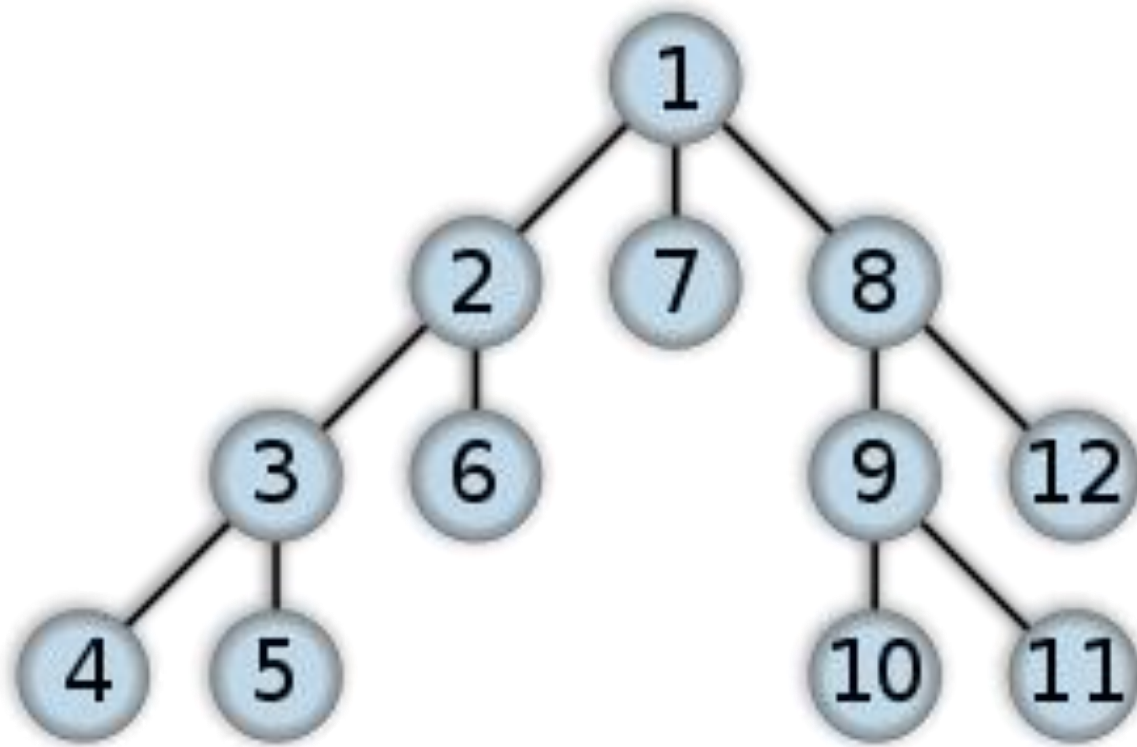
참여방법/구글 meet

- <https://meet.google.com/gpu-mixd-bqa>
- 해당 링크로 접속
- 해당 스터디 진행영상은 참여하지 못하신 분들을 위해 녹화 후 유튜브 채널에 업로드할 예정입니다^^
- 혹시 녹화를 원하지 않는 분들은 말씀해주세요.
- 아래는 영상이 올라갈 주소 입니다.
- <https://www.youtube.com/playlist?list=PL9gVcwpebJSJJ80vNpdrYAhyG6PXxbgq>

시작하기 전에..

- 이번 6주차에는 먼저 **15분**간 개념 설명 뒤에
- **60분** 동안 모두에게 문제 푸는 시간이 주어집니다.
 - 도중에 궁금한 문제나 풀리지 않는 문제가 있다면
슬랙방에 질문을 하시고, **익명**으로도 가능하니 스터디 운영진에게 **Direct Message**를 통해 알려주세요.
- 나머지 **40분**은 풀이 설명을 진행할 것입니다.
 - 시청은 **자유**이며, 여러분이 **해당 문제**의 설명을 듣지 않아도 된다고 판단하면 그 시간에 다른 문제를 푸셔도 좋습니다!



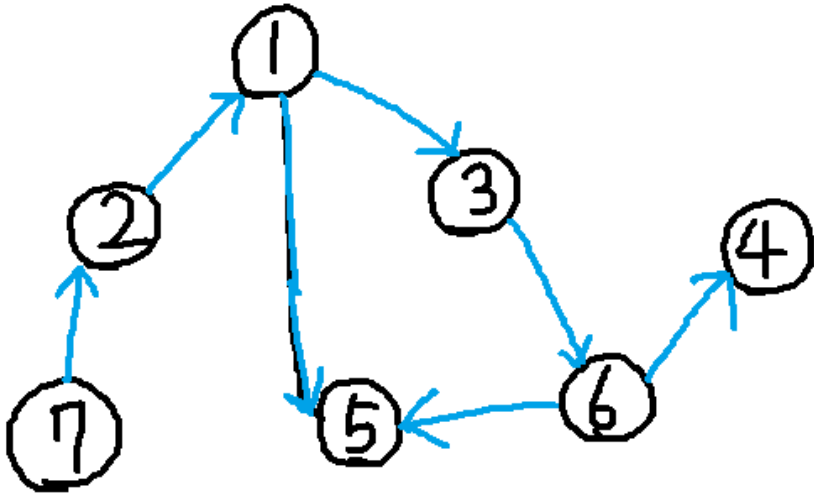


DFS 깊이 우선 탐색

Depth-First Search

2차원 배열로 그래프 간선(u, v) 표현하기

• 유향 그래프

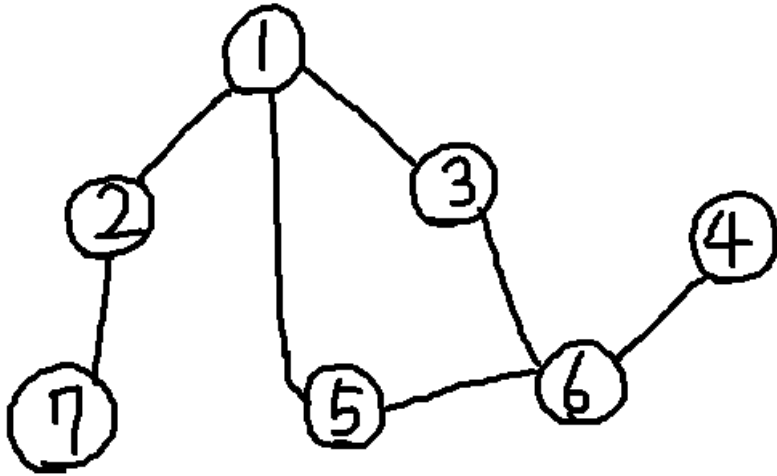


u \ v	1	2	3	4	5	6	7
1			*		*		
2	*						
3						*	
4							
5							
6				*	*		
7		*					

u	v1	v2
1	3	5
2	1	
3	6	
4		
5		
6	4	5
7	2	

2차원 배열로 그래프 간선(u, v) 표현하기

• 무향 그래프



u \ v	1	2	3	4	5	6	7
1		*	*		*		
2	*						*
3	*					*	
4						*	
5	*					*	
6			*	*	*		
7		*					

u	v1	v2	v3
1	2	3	5
2	1	7	
3	1	6	
4	6		
5	1	6	
6	3	4	5
7	2		

노드를 어떻게 취급할 것인가?

- 노드가 **숫자**일 경우
 - 단순히 숫자이므로 그대로 사용하면 됨
- 노드가 **순서쌍**일 경우 (보통 좌표)
 1. `std::pair<int, int>`에 2D 좌표 (y, x)을 넣음
 2. 숫자 하나로 바뀌서 표현
- 어떤 방법을 쓸 지는 자유!
- 2번째 방법을 잠시 살펴봅시다.

2차원 배열과 1차원 배열

2차원 배열을 1차원 배열로 짤푼 늘여봅시다.
가로(열) 길이는 4 입니다.

0	1	2	3
4	5	6	7
8	9	10	11

3 x 4 배열

0	1	2	3	4	5	6	7	8	9	10	11
---	---	---	---	---	---	---	---	---	---	----	----

1 x 12 배열

각 인덱스를 가로 길이로 나누면..

4로 나누었을 때 (몫, 나머지)를 구해봅니다.

0 (0, 0)	1 (0, 1)	2	3
4 (1, 0)	5	6	7
8	9	10	11

3 x 4 배열

0	1	2	3	4	5	6	7	8	9	10	11
---	---	---	---	---	---	---	---	---	---	----	----

1 x 12 배열

눈치채셨나요?

배열 색인과 2차원 배열의 **가로길이**만 알면, 좌표를 구할 수 있습니다!

$y = \text{index} / 4$ (몫)

$x = \text{index} \% 4$ (나머지)

0 (0, 0)	1 (0, 1)	2 (0, 2)	3 (0, 3)
4 (1, 0)	5 (1, 1)	6 (1, 2)	7 (1, 3)
8 (2, 0)	9 (2, 1)	10 (2, 2)	11 (2, 3)

3 x 4 배열

0	1	2	3	4	5	6	7	8	9	10	11
---	---	---	---	---	---	---	---	---	---	----	----

1 x 12 배열

반대로도 가능할까요?

2차원 배열 가로길이와 좌표를 알면, 인덱스를 구할 수 있지요 :)

0 (0, 0)	1 (0, 1)	2 (0, 2)	3 (0, 3)
4 (1, 0)	5 (1, 1)	6 (1, 2)	7 (1, 3)
8 (2, 0)	9 (2, 1)	10 (2, 2)	11 (2, 3)

3 x 4 배열

$$\text{index} = \text{가로 길이} * y + x$$

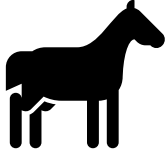
0	1	2	3	4	5	6	7	8	9	10	11
---	---	---	---	---	---	---	---	---	---	----	----

1 x 12 배열

그래프를 완성한 뒤에 dfs를 실행한다

- dfs부터 실행하면 간선을 탐색할 수 없음
 - 당연한 얘기
- 2차원 배열에서 연결할 수 있는 모든 간선을 만든 뒤에 탐색
 - Ex) 노드 a와 b가 상,하,좌,우로 인접한 경우 간선을 이음

좌표 오프셋

	상 (y - 1, x)	
좌 (y, x - 1)	 (y, x)	우 (y, x + 1)
	하 (y + 1, x)	

ox	0	+1	0	-1
oy	-1	0	+1	0

```
for (int i = 0; i < 4; ++i)
{
    new_y = y + oy[i];
    new_x = x + ox[i];
}
```

정점 방문 여부 체크

- 내가 해당 정점을 방문했는지의 여부를 저장하는 데이터
- **좌표** 또는 **노드 정점 번호**와 매핑
- 방문을 이미 했었다면 방문하지 않고,
- 그 반대인 경우는 방문과 동시에 체크인(visited = true)

boj-1012 유기농 배추 (난이도 : 하)

<https://www.acmicpc.net/problem/1012>

boj-1012 유기농 배추

- 가장 기본적인 DFS/BFS탐색 문제입니다.
- 사전에 `dx[],dy[]`를 미리 만들고 상하좌우 탐색을 시도합니다.
- dfs를 사용시, 일반적으로 재귀함수를 많이 사용합니다.
- 각 칸을 살펴보면서 상하좌우로 뻗어 나가게 할려면 어떻게 재귀함수를 짜야할지 고려해봅니다.

boj-1012 유기농 배추

- 각 모든 칸에 대해서 만약 배추이면 `cnt++`, dfs탐색을 합니다.
- 하지만 배추가 아니면 그냥 넘어갑니다.
- 이때 dfs를 하면서 방문표시를 해주어 재방문을 하지 않도록 합니다.
- 모든 칸에 대해서 배추밭을 만난 횟수를 새어주면 정답이 됩니다.(한번 방문한 칸은 방문표시를 해주었기 때문에)

boj-1012 유기농 배추

- https://github.com/alps-jbnu/ALPS_2020_Summer_Study/blob/master/KiwanKim/1012.c
[pp](#)

boj-2573 빙산 (난이도 : 중상)

<https://www.acmicpc.net/problem/2573>

boj-2573 빙산

- 기본적인 dfs풀이법으로 접근하되 어떤 것을 추가로 고려해야 하는지 고려해봅니다.
- 어떻게 dfs를 시도해야 시간초과가 나지 않고 효율적으로 dfs를 하면서 녹아지는 지도를 그릴 수 있을지 고려해봅니다.

boj-2573 빙산

- 모든 칸에 대해서 0이 아닌 부분(섬)을 만나면 dfs를 들어갑니다.
- dfs함수 내에서 상하좌우를 확인하되 또다른 섬을 만나면 다시 dfs를 들어가주고 반면 0(바다)를 만났을 때는 count합니다.
- apply_melt()에서 0(바다)를 count한 배열을 통해 녹은 지도를 완성합니다.

boj-2573 빙산

- https://github.com/alps-jbnu/ALPS_2020_Summer_Study/blob/master/KiwanKim/2573.cpp
- <https://jaimemin.tistory.com/653>

boj-1194 달이 차오른다, 가자.
(난이도 : 상)

<https://www.acmicpc.net/problem/1194>

boj-1194 달이 차오른다, 가자.

- 기존 dfs탐색문제처럼 방문상태를 저장하고 업데이트해가는 방식으로 풀어갑니다.
- 열쇠를 가진 상태별로 비트마스크를 통해 표시합니다.(열쇠 6개 $\Rightarrow 2^6$)
- dfs를 통해 각각 다른 상태별 칸에 대해서 어떻게 대처할지 고려해봅니다.

boj-1194 달이 차오른다, 가자.

- `cnt[N][M][2^6(열쇠의 상태)]`로 구성합니다.
- 열쇠칸, 문칸, 일반칸, 정답칸을 구분하고 정답칸(1)을 만날때 까지 dfs를 돕니다.
- 열쇠칸을 만나게 되면 비트연산 및 `cnt`를 업데이트 합니다.
- 문칸을 만났을 경우 비트연산을 통해 열쇠를 갖고 있는지 체크합니다. 갖고 있다면 다시 dfs를 돕니다.
- 일반칸을 만났다면 비트연산없이 `cnt`만 업데이트하고 dfs를 돕니다.

boj-1194의 비트마스크

- 000000으로 시작하고 각 칸은 열쇠6개의 소유함을 표시합니다.
- 만약 a열쇠를 가지고 있다면 $000000 \mid 000001 \Rightarrow 000001$ 과 같은 연산을 수행합니다.(OR연산)
- 만약 문A를 만났다면 $000000 \& 000001 \Rightarrow 000000$ 과 같은 연산을 수행합니다.(AND연산)
- 만약 열쇠A와 문A가 만난다면 $000001(\text{열쇠}) \& 000001(\text{문}) \Rightarrow 000001$ 와 같이 0보다 큰수 나옵니다.
- 문과 만났을때, 0이 나오면 "열쇠가 없다", 0보다 큰수가 나오면 "열쇠가 있다"

boj-1194의 비트마스크

- 코드상의 비트마스크
- `int addKeys = keys | (1 << (Map[tx][ty] - 'a'));`
- 기존 keys에 대해서 해당 열쇠의 번호만큼 1을 SHIFT한 수를 OR연산
- Ex) `0000000 | (1 << 3)` 은 `0000000 | 0001000 => 0001000`
- `if((keys & (1 << (Map[tx][ty] - 'A')))) == 0)`
- 기존의 keys에 대해서 해당 열쇠의 번호 SHIFT한 수를 AND연산
- Ex) `001010 & (1 << 4)` 은 `001010 & 001000 => 001000`
- 반면 일치하는 열쇠가 없으면 0이 나온다.

boj-1194 달이 차오른다, 가자.

- https://github.com/alps-jbnu/ALPS_2020_Summer_Study/blob/master/KiwanKim/1194.cpp
- <https://jaimemin.tistory.com/773>(BFS)

boj-2468 안전 영역 (난이도 : 하)

<https://www.acmicpc.net/problem/2468>

boj-2468 안전 영역

- 수면이 높이 **0 ~ 100**까지 차오르는 경우를 완전 탐색
- 조금만 최적화하면?
 - 수면이 높이 **0 ~ 가장 높은 지대**까지 차오르는 경우 완전 탐색
- 잠기는 영역은 DFS로 풀기

boj-2468 안전 영역

- DFS로 연결된 지대 구하기
- 수면 높이가 1씩 올라갈 때마다 정보가 업데이트 되어야 하므로
 - 그래프 방문 여부를 false로 설정
 - 해당 수면 높이에서 물에 잠기는 지대를 업데이트
 - 현재 지대에서 상하좌우 1칸에 있는 지대를 방문할 수 있는지 따짐
 - 방문할 수 있다면 dfs 실행

boj-2468 안전 영역

- 발상의 전환
 - 지대가 수면에 잠기는지의 여부를 visited로 대신할 수 없을까요?
 - 가능합니다!
 - 미리 방문처리를 하면 됩니다 `visited = true`

boj-2468 안전 영역

- https://github.com/alps-jbnu/ALPS_2020_Summer_Study/blob/master/Jubin/week6/boj-2468.cpp

programmers-43163 단어 변환 (난이도 : 중)

<https://programmers.co.kr/learn/courses/30/lessons/43163>

programmers-43163 단어 변환

- 정점을 번호 대신 문자열을 사용해서 풀 수 있습니다.
- 문자열 $s1$, $s2$ 가 상호 단어 변환이 가능한지 판단하는 함수 작성
- $target$ 이 $words$ 에 들어있지 않으면?
 - 애초에 단어 변환이 불가능

programmers-43163 단어 변환

- 사실 꼼수를 쓰면 정점을 번호로 놓아도 풀 수 있습니다.
 - 입력된 데이터 words 위치가 고정되어 있으므로!

programmers-43163 단어 변환

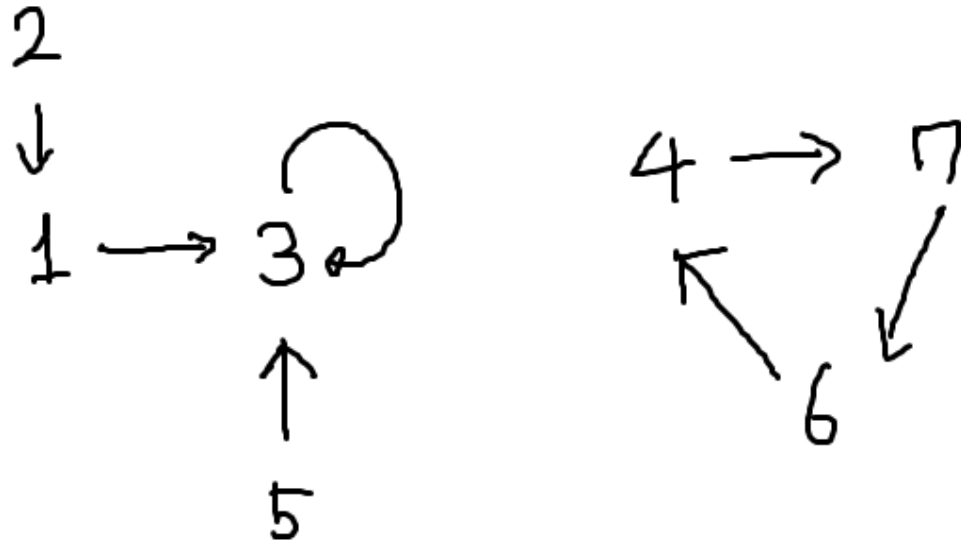
- https://github.com/alps-jbnu/ALPS_2020_Summer_Study/blob/master/Jubin/week6/programmers-43163.cpp

boj-9466 팀 프로젝트 (난이도 : 상)

<https://www.acmicpc.net/problem/9466>

boj-9466 텀 프로젝트

- 유형 그래프
- 방문 시작 여부와 방문 끝 여부를 판정
 - 방문을 이전에 시작했었지만 끝나지 않은 경우엔 사이클을 탐색
 - 방문을 안 했다면 인접 노드를 dfs



boj-9466 텀 프로젝트

- https://github.com/alps-jbnu/ALPS_2020_Summer_Study/blob/master/Jubin/week6/boj-9466.cpp

다음 7주차에는..

- BFS를 배웁니다.
- 감사합니다.