

# 알고리즘 특강 완전탐색

모든 경우의 수를 탐색하는 기법입니다. 당연히 아이디어는 단순하지만, 실제로 많이 사용되기 때문에 한번쯤 고민해봐야 합니다.

### 완전탐색



#### **Brute Force**

- 단순히 가능한 모든 경우를 확인하는 기법.
- 테스트 케이스의 크기에 따라 소요 시간이 엄청나게 커질 수 있음.
- 그러나, 떠올릴 수 있는 <mark>가장 쉬운 방법</mark>이기 때문에, 문제를 풀 때 가장 먼저 고민해야 하는 방법이기도 함.
- 표본의 수와 시간복잡도를 전체적으로 고려해 선택하는 것이 아주 중요한 기법.





# 100,000,000

- 총 연산수가 약 1억 회 이하인 경우, 충분히 <mark>완전탐색으로 접근할 수 있음</mark>.
- 실제와는 거리가 있지만, 대략적인 완전탐색 가능 기준으로 잡으면 유용함.



### 간단한 예시부터..



### Silver 5 - 영화감독 숌

### 요약

- 종말의 숫자란 어떤 수에 6이 적어도 3개 이상 연속으로 들어가는 수를 말한다.
- 즉, 666, 1666, 2666, 3666 … 은 종말의 숫자이다.
- 이때, N번째 종말의 숫자를 구하는 프로그램을 작성하시오.

# 제약조건

• N의 범위는 1 <= N <= 10,000 이다.





✓ Silver 5 - 영화감독 숌

#### 요약

- 종말의 숫자란 어떤 수에 6이 적어도 3개 이상 연속으로 들어가는 수를 말한다.
- 즉, 666, 1666, 2666, 3666 … 은 종말의 숫자이다.
- 이때, N번째 종말의 숫자를 구하는 프로그램을 작성하시오.

### 제약조건

• N의 범위는 1 <= N <= 10,000 이다.

### 접근

- 666 앞과 뒤에 숫자를 계속 붙이고, 정렬해볼까?
- → 충분히 가능. 다른 방법은 없을까?





#### ✓ Silver 5 - 영화감독 숌

#### 요약

- 종말의 숫자란 어떤 수에 6이 적어도 3개 이상 연속으로 들어가는 수를 말한다.
- 즉, 666, 1666, 2666, 3666 … 은 종말의 숫자이다.
- 이때, N번째 종말의 숫자를 구하는 프로그램을 작성하시오.

### 제약조건

• N의 범위는 1 <= N <= 10,000 이다.

### 접근

- 666 앞과 뒤에 숫자를 계속 붙이고, 정렬해볼까?
- → 충분히 가능. 다른 방법은 없을까?
- 1부터 수를 계속 올리면서 666 있는지 판단 하는 건?
- ---- 10000번째 종말의 숫자는 6,669,999 보다 작음. (Why? 6,660,000 ~ 6,669,999: 1만개)
- 6,669,999 < 100,000,000 이므로,</li>충분히 루프를 돌려 해결할 수 있음.
- 다른 풀이가 있을 수 있지만,
  - 완전탐색 풀이가 더 간단한 경우가 많음.
  - 시험장에서 다른 풀이가 떠오르지 않을 수 있음.
- 완전탐색을 먼저 떠올려 보는 것이 유리하다!





### ✓ Silver 4 - 퇴사

### 요약

- N + 1번째 날 퇴사를 하기 위해 N일 동안 최대한 많은 상담을 하려고 한다.
- 1~N일 까지 매일 상담 스케쥴이 있는데, 각각의 상담은 상담을 완료하는데 걸리는 시간 T와 보수 P가 존재한다.
- 상담을 적절히 했을 때, 얻을 수 있는 최대 수익을 구하는 프로그램을 작성하시오.

# 제약조건

- 퇴사까지 남은 날은 1 <= N <= 15 이다.
- 시간 T<sub>i</sub>와 P<sub>i</sub>의 범위는 각각 1 <= T<sub>i</sub> <= 5, 1 <= P<sub>i</sub> <= 1,000 이다.



### 결국은 구현이 중요.



#### Silver 3 - 오목

### 요약

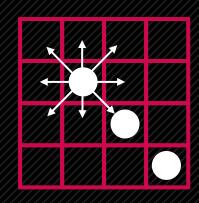
- 오목은 흑과 백이 번갈아 돌을 두고, 가로/세로/대각선으로 5개의 돌을 연속해서 두면 승리하는 게임이다.
- 다만, 6개 이상의 돌이 한번에 이어졌을 때는 승리로 인정하지 않는다.
- 바둑판의 상태가 주어졌을 때, 검은색이 이겼는지, 흰색이 이겼는지, 둘 다 아닌지 판단하는 프로그램을 작성하시오.

# 제약조건

• 19 \* 19 오목판이 주어지며, 숫자는 한 칸 씩 띄어서 표시된다.



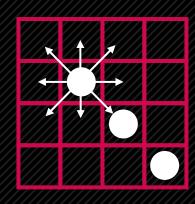




● 모든 칸에 대하여, 5개 나올 때 까지 8방향으로 탐색.



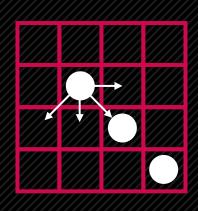




● 꼭 8 방향을 다 체크해야 할까?



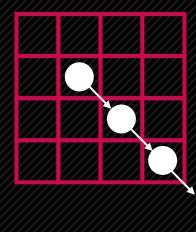




- 꼭 8 방향을 다 체크해야 할까?
- → 4방향만 체크하면 OK!



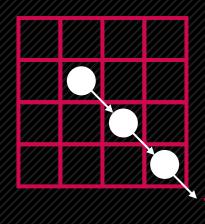




● 진행하다 중간에 막히게 된다면?



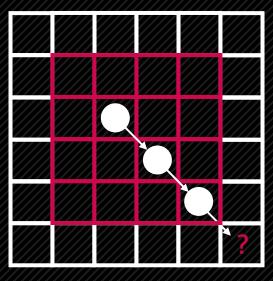




- 진행하다 중간에 막히게 된다면?
- → 전진 후, 끝까지 왔는지 체크!



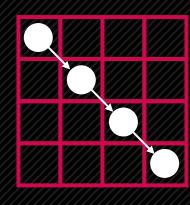




- 진행하다 중간에 막히게 된다면?
- → 처음부터 둘레를 만들어도 된다!



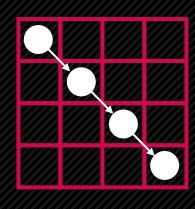




● 6칸이 넘는 건 어떻게 체크할까?





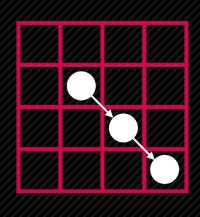


- 6칸이 넘는 건 어떻게 체크할까?
- → 5칸 전진 후, 한 칸 더 전진!



### 결국은 구현이 중요.

### 접근



- 꼭 8 방향을 다 체크해야 할까?
- 진행하다 중간에 막히게 된다면?
- 6칸이 넘어가는 건 어떻게 체크할까?

구현 문제를 풀 땐, 발생할 수 있는 문제/예외에 대해 미리 분석하고 푸는 것이 중요!



### 실습

### ✓ Silver 3 - 전공책

### 요약

- 전공책의 제목을 이용해 새로운 글자를 만들려고 한다.
- 만들려고 하는 텍스트와 전공책의 제목, 가격이 주어졌을 때, 필요한 최소한의 비용을 출력하는 프로그램을 작성하시오.

### 제약조건

- 만들고자 하는 문자열의 길이 T는 1<= T <= 10 이다.
- 전공책의 수 N은 1 <= N <= 16 이다.
- I번째 전공책의 가격 Ci는 10,000 <= Ci <= 100,000 이다.
- I번째 전공책의 제목의 길이 W<sub>I</sub>는 1 <= W<sub>I</sub> <= 50 이다.



### 추가 추천 문제 #1

- ✓ Silver 4 한수
  - 문제를 차근차근 분석해 봅시다.
- ★ Silver 3 마인크래프트
  - B가 크다고 놀라지 마세요. 어차피 B의 범위는 우리의 풀이과정에 별로 도움이 되지 않습니다.
- ✓ Silver 3 토너먼트
  ★ Level 2 예상 대진표
  - 약간의 수학이 필요합니다.
- Level 2 카펫
  - 변수가 두개라서 어떻게 완전 탐색을 할지 감이 안 오시나요? 두 변수를 하나로 묶을 수 있는 방법이 없을까요?





### Backtracking

- 완전탐색처럼 모든 경우를 탐색하나, 중간 중간에 조건에 맞지 않는 케이스를 가지치기 하여 탐색 시간을 줄이는 기법.
- 일반적으로 완전탐색에 비해 시간적으로 효율적임.
- 탐색 중 조건에 맞지 않는 경우 이전 과정으로 돌아가야 하기 때문에, 재귀를 사용하는 경우가 많음.
- 조건을 어떻게 설정하고, 틀렸을 시 어떤 시점으로 돌아가야 할지 설계를 잘 해야 함.





Silver 3 - N과 M (1)

요약

• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 <mark>중복없이 M개를 고른 수열</mark>을 모두 구하는 프로그램을 작성하시오.

제약조건

• N과 M의 범위는 1<= M <= N <= 8 이다.

### 문제를 한 번 봅시다…



//> Silver 3 - N과 M (1)

요약

• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 중복없이 M기를 고른 수열을 모두 구하는 프로그램을 작성하시오.

제약조건

• N과 M의 범위는 1 <= M <= N <= 8 이다.

### 접근

- 어떤 제약 조건이 있을까?
- → 수열 내부에는 <mark>중복이 있으면 안된</mark>다.
- → 수열 크기는 M이어야 한다.

#### 문제를 한 번 봅시다…



/<> Silver 3 - N과 M (1)

요약

• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 중복없이 M개를 고른 수열을 모두 구하는 프로그램을 작성하시오.

제약조건

• N과 M의 범위는 1 <= M <= N <= 8 이다.

### 접근

- 어떤 제약 조건이 있을까?
- 수열 내부에는 중복이 있으면 안된다.
- → 수열 크기는 M이어야 한다.
- 재귀함수를 설계 해보자.
- → 각수를 넣으려고 할때, 이미 수열내에 있으면 스킵.
- → 수열 크기가 M이 되면 리턴.





✓ Silver 3 - N과 M (2)



- 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 <mark>중복없이 M개를 고른 수열을 모두</mark> 구하는 프로그램을 작성하시오.
- 단, 수열은 오름차순이다.

# 제약조건

• N과 M의 범위는 1 <= M <= N <= 8 이다.

### 1과 무슨 차이가 있을까?



### 접근

- 어떤 제약 조건이 있을까?
- 수열 내부에는 중복이 있으면 안된다.
- 수열 크기는 M이어야 한다.
- 재귀함수를 설계 해보자.
- → 각 수를 넣으려고 할 때, 이미 수열 내에 있으면 스킵.
- → 수열 크기가 M이 되면 리턴

단, 수열은 오름차순이다.

?

### 1과 무슨 차이가 있을까?



### 접근

- 어떤 제약 조건이 있을까?
- 수열 내부에는 중복이 있으면 안된다.
- 수열 크기는 M이어야 한다.
- 재귀함수를 설계 해보자.
- → 각수를 넣으려고 할 때, 이미 수열 내에 있으면 스킵.
- → 수열 크기가 M이 되면 리턴.

단, 수열은 <mark>오름차순</mark>이다.

- 오름차순은 어떻게 걸러낼까?
- → 마지막 원소를 확인하여(해당 값 + 1) 부터 반복문 수행





✓ Silver 3 - N과 M (3)

요약

• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 <mark>중복을 포함하여 M개를 고른 수열</mark>을 모두 구하는 프로그램을 작성하시오.

제약조건

• N과 M의 범위는 1 <= M <= N <= 8 이다.





✓ Silver 3 - N과 M (4)



- 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 <mark>중복을 포함하여 M개를 고른 수열</mark>을 모두 구하는 프로그램을 작성하시오.
- 단, 수열은 비내림차순이다.

### 제약조건

• N과 M의 범위는 1 <= M <= N <= 8 이다.









✓ Gold 5 (Level 3) - N-Queen



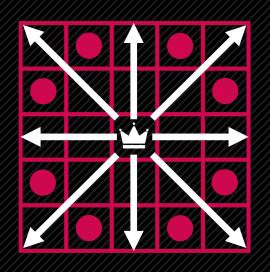
• N \* N 인 체스판 위에 퀸 N개가 서로 공격할 수 없게 놓는 경우의 수를 구하는 프로그램을 작성하시오.



• N의 범위는 1 < N <= 15 이다.

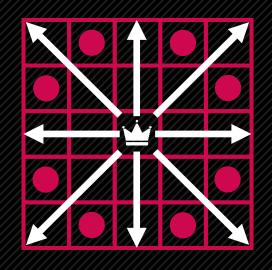






● 퀸이 놓이면, 가로, 세로, 대각선에 위치한 곳엔 말을 둘 수 없다.

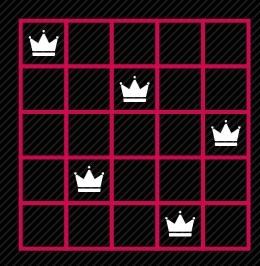




퀸이 놓이면, 가로, 세로, 대각선에 위치한 곳엔 말을 둘 수 없다.

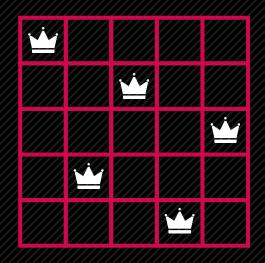
- 완전탐색은 어떨까?
- → <sub>255</sub>C<sub>15</sub> > 10<sup>30</sup>. 택도 없다!
- 가지치기 할 방법은 없을까?
- 어차피 각각 가로와 세로에는 1개만 들어감.





● 어차피 각각 가로와 세로에는 1개만 들어감.





어차피 각각 가로와 세로에는 1개만 들어감.

✓ Silver 3 - N과 M (1)

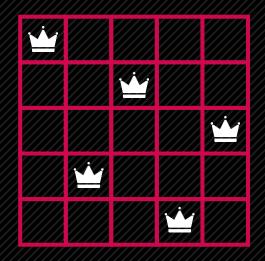


• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 중복없이 M기를 고른 수열을 모두 구하는 프로그램을 작성하시오.



• N과 M의 범위는 1 <= M <= N <= 8 이다.





어차피 각각 가로와 세로에는 1개만 들어감.

✓ Silver 3 - N과 M (1)



• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 중복없이 M개를 고른 수열을 모두 구하는 프로그램을 작성하시오.



- N과 M의 범위는 1 <= M <= N <= 8 이다.
- 결국, 시도할 수 있는 경우는 N! 그럼에도 15! > 10<sup>11...</sup>
- 하지만, 대각선 가지치기를 하면 충분히 걸러 낼 수 있을 것!



Silver 3 - N과 M (1)



• 자연수 N과 M이 주어졌을 때, 1부터 N까지 자연수 중 <mark>중복없이 M개를 고른 수열</mark>을 모두 구하는 프로그램을 작성하시오.

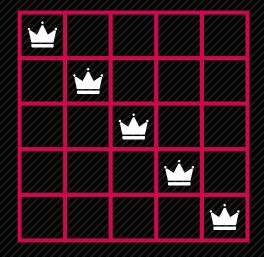


• N과 M의 범위는 1 <= M <= N <= 8 이다.

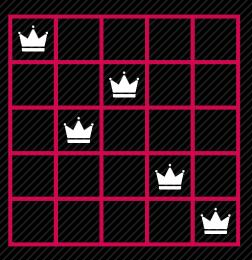
1, 2, 3, 4, 5, 6, 7, 8 1, 3, 2, 4, 5, 6, 7, 8

---



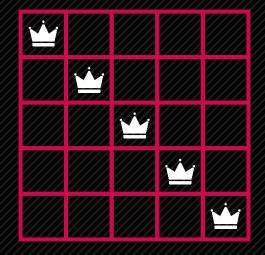


1, 2, 3, 4, 5

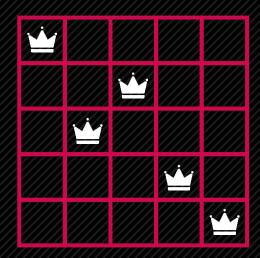


1, 3, 2, 4, 5





if(abs(arr[i] - arr[j]) == j - i)



1, 2, 3, 4, 5

1, 3, 2, 4, 5

### 익숙한 문제죠?



### ✓ Silver 3 - 모든 순열



• N이 주어졌을 때, 1부터 N까지의 수로 이루어진 순열을 사전순으로 출력하는 프로그램을 작성하시오.



• N의 범위는 1 <= N <= 8 이다.



#### Itertools!

```
import itertools
iter = itertools.combinations('12345678', 5)
for data in iter:
    print(' '.join(map(str, data)))
```

- Iterable한 데이터들의 조합/순열/합성곱을 쉽게 구해주는 라이브러리.
- 모든 경우를 고려해야 하는 문제에서 상당히 유용하게 사용됨!



#### Itertools!

```
import itertools
                                                                                                 Itertools 를 미리 import 하고 사용.
                                                                                                   combinations: 주어진 데이터 중 M개를 뽑아
iter = itertools.combinations('1234', 2)
# 12 13 14 23 24 34
                                                                                                    가능한 모든 조합 리턴
for data in iter:
   print(' '.join(map(str, data)))
                                                                                                    permutations: 주어진 데이터 중 M개를 뽑아
iter = itertools.permutations('1234', 2)
                                                                                                    가능한모든 <mark>순열</mark> 리턴
# 12 13 14 21 23 24 31 32 34 41 42 43
for data in iter:
   print(*data)
                                                                                                    combinations_with_replacement
iter = itertools.combinations_with_replacement('1234', 2)
                                                                                                    주어진데이터중<mark>중복을 포함</mark>하여M개를 뽑아
# 11 12 13 14 22 23 24 33 34 44
for data in iter:
                                                                                                    가능한 모든 조합 리턴
   x, y = data
   print(x, y)
                                                                                                   combinations_with_replacement:
주어진 데이터 목록의 Catesian Product 리턴
iter = itertools.product('1234', repeat=2)
# 11 12 13 14 21 22 23 24 31 32 33 34 41 42 43 44
for data in iter:
   print('{} {}'.format(*data))
```





### ✓ Silver 3 - 스타트와 링크 (삼성 SW 역량테스트 기출)

# 요약

- 짝수 인원 N명이 있을 때, N/2 명으로 구성된 두 팀을 만들려고 한다.
- I번째 사람과 J번째 사람이 같은 팀을 했을 시, 얻을 수 있는 능력치는 S<sub>J</sub> + S<sub>J</sub> (S<sub>J</sub> 와 S<sub>J</sub> 는 다를 수 있음.) 이다.
- 게임의 밸런스를 위해, 주어진 능력치를 기반으로 두 팀의 능력치의 합을 최소로 하려고 한다.
- 차이의 최솟값을 출력하는 프로그램을 작성하시오.

# 제약조건

- N의 범위는 1 <= N <= 20 이다.
- S<sub>II</sub>는 항상 0이며, S<sub>II</sub>의 범위는 1 <= S<sub>II</sub> <= 100 이다.



### 추가 추천 문제 #2

- ⋉ Silver 2 계란으로 계란치기
  - 문제가 아주 길지만 결국 앞에서 배운 내용이 전부입니다. 긴 문제를 읽는 연습이라고 생각해봅시다.
- ✓ Silver 1 연산자 끼워넣기 (삼성 SW 역량테스트 기출)
  - 구현에 집중된 문제지만, 결국 구현 과정에서 배운 내용을 사용해야 합니다.
- - 많이 생각해야 하는 문제입니다. 어떤 방식으로 백트래킹을 시도해야 할까요?
- ★ Level 2 소수 찾기
  - 순서와 길이를 모두 고민해서 숫자를 추출해야 합니다. 어떤 방법으로 하면 좋을까요?

</s>

"Any question?"