Chapter. 02

알고리즘

완전 탐색 (brute force)

✓ 연습 문제

FAST CAMPUS
ONLINE
알고리즘 공채 대비반I

강사. 류호석

Life Changing Education

Chapter. 02

알고리즘
완전 탐색(Brute Force)

FAST CAMPUS
ONLINE
류호석 강사.

Fast campus

I 완전탐색 - 총 정리

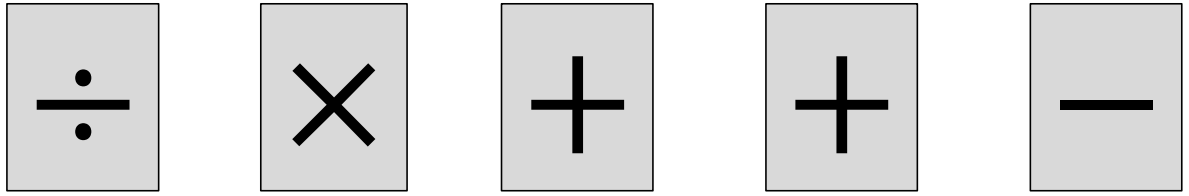
중복	순서	시간 복잡도	공간 복잡도
YES	YES	$O(N^M)$	$O(M)$
NO	YES	$O({}_M^N P) = O\left(\frac{N!}{(N-M)!}\right)$	$O(M)$
YES	NO	$O(N^M)$ 보단 작음	$O(M)$
NO	NO	$O({}_M^N C) = O\left(\frac{N!}{M!\,(N-M)!}\right)$	$O(M)$

I BOJ 14888 – 연산자 끼워넣기

난이도: 2

N=6 => 6 2 3 5 3 4

N-1 개의 연산자

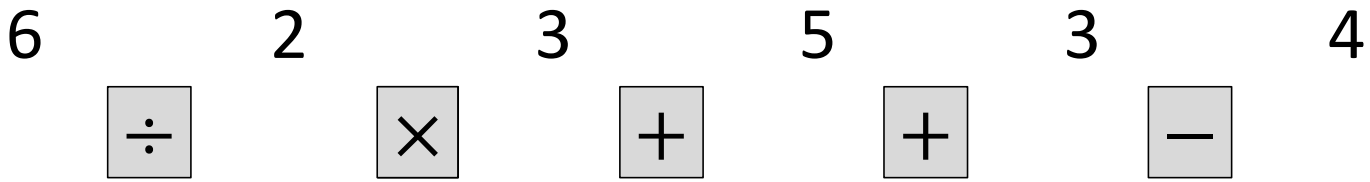


사칙연산 => [+, −, ×, ÷]

cnt[x] := x 번째 사칙연산이 몇 개 사용 가능한 지

I BOJ 14888 – 연산자 끼워넣기

난이도: 2



$$6 \div 2 \times 3 + 5 + 3 - 4 =$$
$$6 \times 2 + 3 \div 5 - 3 + 4 =$$
$$6 + 2 - 3 \times 5 \div 3 + 4 =$$
$$\cdots$$

I 문제 파악하기 – 정답의 최대치

출력

첫째 줄에 만들 수 있는 식의 결과의 최댓값을, 둘째 줄에는 최솟값을 출력한다.

연산자를 어떻게 끼워넣어도 항상 -10억보다 크거나 같고, 10억보다 작거나 같은 결과가 나오는 입력만 주어진다.

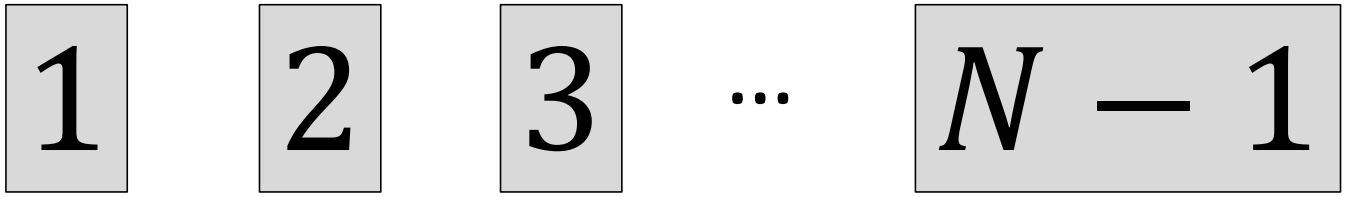
또한, 앞에서부터 계산했을 때, 중간에 계산되는 식의 결과도 항상 -10억보다 크거나 같고, 10억보다 작거나 같다.

int 범위: -21억 ~ 21억

→ int 형 변수를 쓰면 된다!

BOJ 14888 – 연산자 끼워넣기

난이도: 2



N-1 개의 카드 중에서 **중복 없이**(같은 카드는 한 번 사용해서)
N-1 개를 **순서 있게 나열하기**

중복	순서	시간 복잡도	공간 복잡도
YES	YES	$O(N^M)$	$O(M)$
NO	YES	$O({}_N^M\mathrm{P}) = O\left(\frac{N!}{(N-M)!}\right)$	$O(M)$
YES	NO	$O(N^M)$ 보단 작음	$O(M)$
NO	NO	$O\left(\binom{N}{M}\right) = O(\frac{N!}{M!\,(N-M)!})$	$O(M)$

10! = 3628800

I 구현 스케치

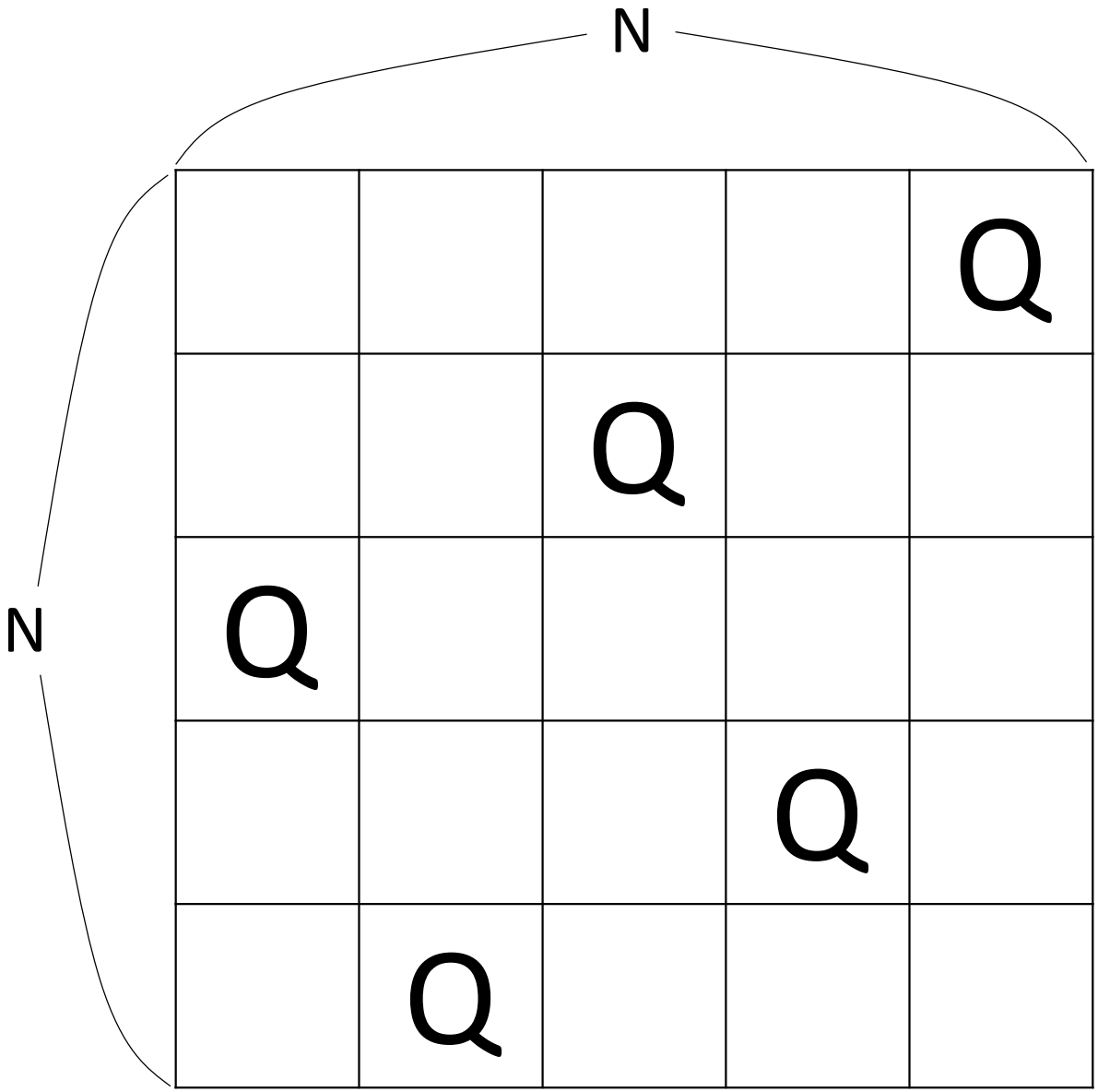
```
static int N, max, min;
static int[] nums, operators, order;

// order[1...N-1] 에 연산자들이 순서대로 저장될 것이다.
static void rec_func(int k) {
    if (k == N) {  // 모든 연산자들을 전부 나열하는 방법을 찾은 상태

        // 정한 연산자 순서대로 계산해서 정답을 갱신하기

    } else {  // k 번째 연산자는 무엇을 선택할 것인가?

        // 4 가지의 연산자들 중에 뭘 쓸 것인지 선택하고 재귀호출하기

    }
}
```

I 구현 스케치 - 심화

```
static int N, max, min;
static int[] nums, operators, order;

// order[1...N-1] 에 연산자들이 순서대로 저장될 것이다.
// k-1 번째 연산자까지 계산한 결과가 value 이다.
static void rec_func(int k, int value) {
    if (k == N) {  // 모든 연산자들을 전부 나열하는 방법을 찾은 상태

        // value 를 정답에 갱신해준다.

    } else {  // k 번째 연산자는 무엇을 선택할 것인가?

        // 4 가지의 연산자들 중에 뭘 쓸 것인지 선택하고 연산자를 계산한 후에 재귀호출하기

    }
}
```

I BOJ 9663 – N Queen

난이도: 2



FAST CAMPUS
ONLINE
류호석 강사.

I 문제 파악하기 – 정답의 최대치

출력

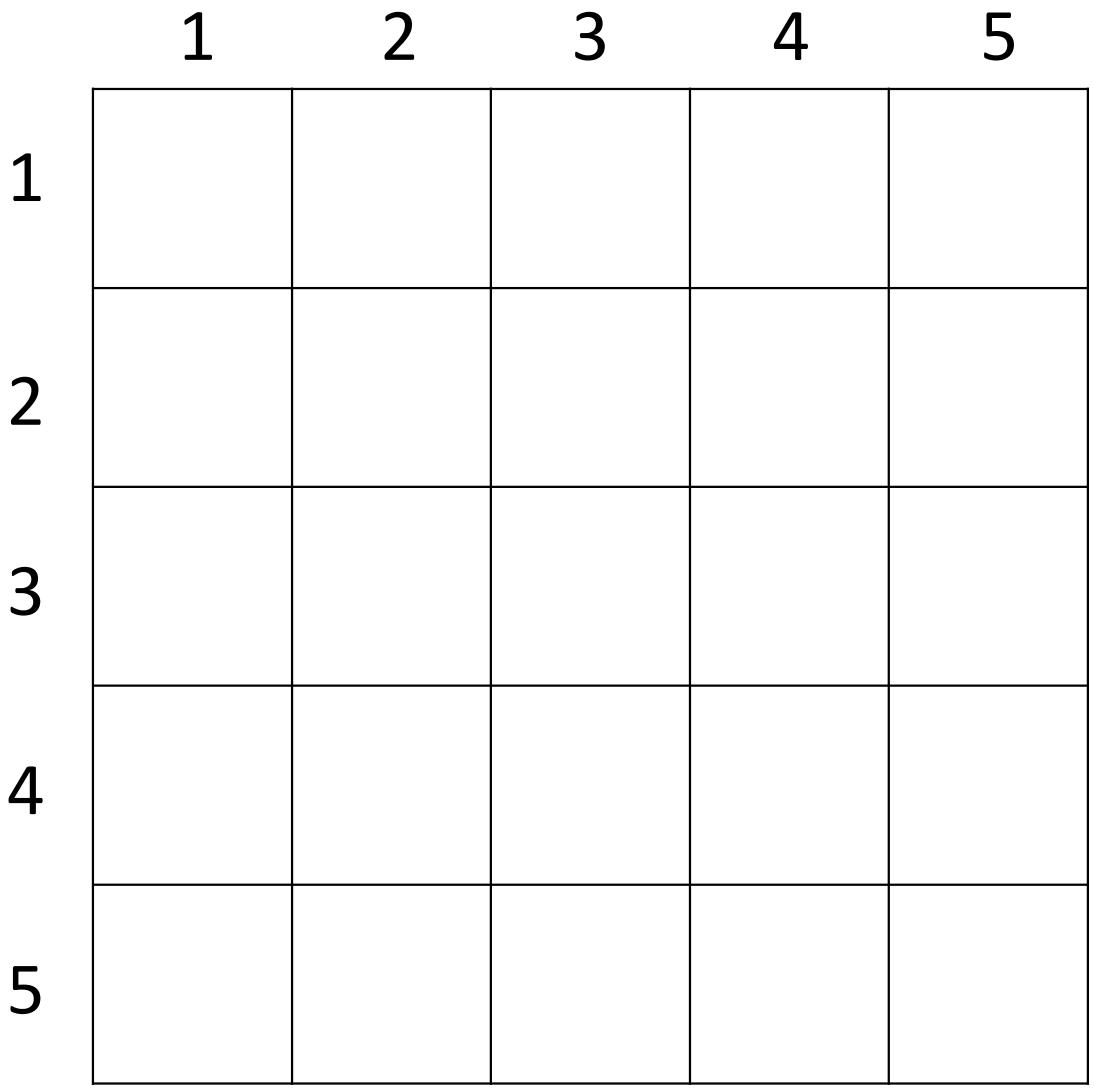
첫째 줄에 퀸 N개를 서로 공격할 수 없게 놓는 경우의 수를 출력한다.

N=14 일 때 정답?
21억을 넘는 지 모른다

→

일단 int 로 정하고
N=14 를 입력으로 넣어
보고 확인하기

I 문제 접근하기



FAST CAMPUS
ONLINE
류호석 강사.

I BOJ 9663 – N Queen

N개 중에서 **중복을 허용해서**
N 개를 **순서대로 나열하는** 모든 경우 탐색하기

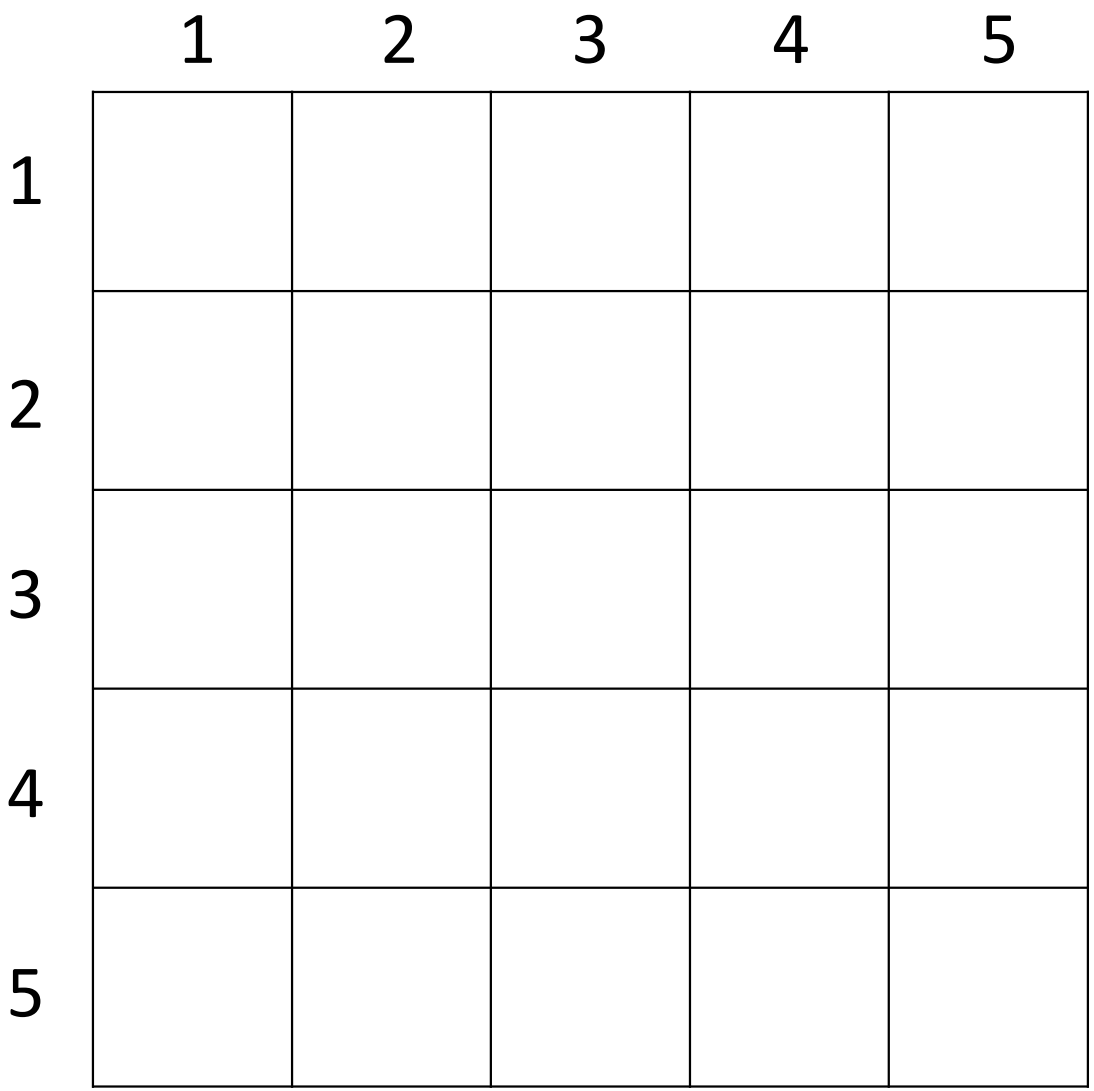
중복	순서	시간 복잡도	공간 복잡도
YES	YES	$O(N^M)$	$O(M)$
NO	YES	$O({}_N\mathrm{P}_M) = O\left(\frac{N!}{(N-M)!}\right)$	$O(M)$
YES	NO	$O(N^M)$ 보단 작음	$O(M)$
NO	NO	$O\left(\binom{N}{M}\right) = O(\frac{N!}{M!\,(N-M)!})$	$O(M$ß$)$

14^14 > 10^16

I 구현

```
static int N, ans;
static int[] col;  // col[i] : i번 행의 퀸은 col[i]번 열에 놓았다는 기록
// row 번 ~ N 번 행에 대해서 가능한 퀸을 놓는 경우의 수 구하기
static void rec_func(int row) {
    if (row == N + 1) {  // 각 행마다 하나씩 잘 놓았다.
        if (validity_check()){  // 서로 공격하는 퀸들이 없는 경우
            ans++;
        }
    } else {
        for (int c = 1; c <= N; c++) {
            col[row] = c;
            rec_func( row: row + 1);
            col[row] = 0;
        }
    }
}
```

I 문제점!

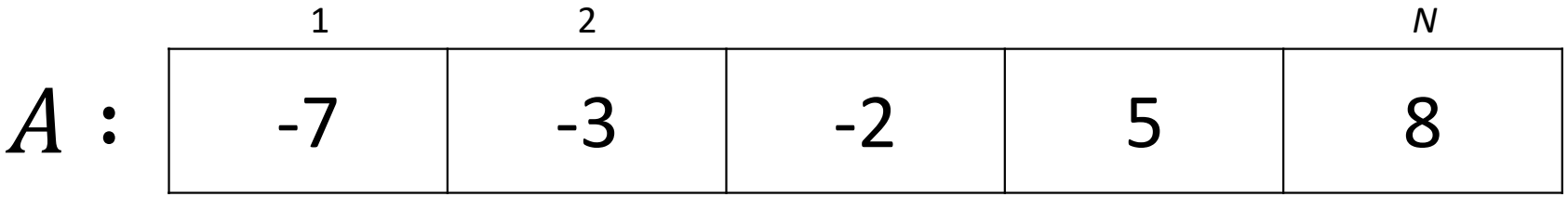


FAST CAMPUS
ONLINE
류호석 강사.

I 구현 스케치 - 심화

```java
static int N, ans;
static int[] col;  // col[i] : i번 행의 퀸은 col[i]번 열에 놓았다는 기록
// row 번 ~ N 번 행에 대해서 가능한 퀸을 놓는 경우의 수 구하기
static void rec_func(int row) {
    if (row == N + 1) {  // 1 ~ N 번 행에 대해서 성공적으로 놓았다!
        ans++;
    } else {
        for (int c = 1; c <= N; c++) {
            // row 행의 c 열에 놓을 수 있으면
            col[row] = c;
            rec_func( row: row + 1);
            col[row] = 0;
        }
    }
}
```

BOJ 1182 – 부분 수열의 합



목표 $S = 0$

부분 수열: 수열의 일부 항을 선택해서 원래 순서대로 나열

진 부분 수열들 중에서 합이 정확히 S 가 되는 경우의 수

I 문제 파악하기 – 정답의 최대치

N <= 20

$|S|$ <= 1,000,000

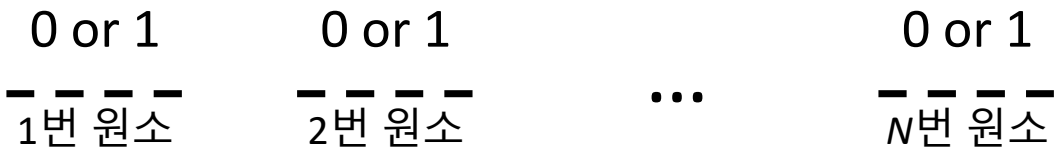
$|A_i|$ <= 1,000,000

부분 수열의 개수 상한: 2^{20} <= 1,048,576 → 정답 변수는 int 형 변수를 쓰면 된다!

부분 수열의 합 상한: 20*1,000,000 → 합을 기록하는 변수는 int 형 변수를 쓰면 된다!

BOJ 1182 – 부분 수열의 합

난이도: 2



0: 부분 수열에 포함시키지 않는다.
1: 부분 수열에 포함시킨다.

중복	순서	시간 복잡도	공간 복잡도
YES	YES	$O(N^M)$	$O(M)$
NO	YES	$O({}_M^N\mathrm{P}) = O\left(\frac{N!}{(N-M)!}\right)$	$O(M)$
YES	NO	$O(N^M)$ 보단 작음	$O(M)$
NO	NO	$O\left(\binom{N}{M}\right) = O(\frac{N!}{M!\,(N-M)!})$	$O(M)$

2^{20} ≅ 100만

I 구현 스케치

```
static int N, S, ans;
static int[] nums;
// k번째 원소를 포함시킬 지 정하는 함수
// value:= k-1 번째 원소까지 골라진 원소들의 합
static void rec_func(int k, int value) {
    if (k == N + 1) {  // 부분 수열을 하나 완성 시킨 상태
        // value 가 S 랑 같은 지 확인하기
    } else {
        // k 번째 원소를 포함시킬 지 결정하고 재귀호출해주기
    }
}
```

I 구현 스케치


```java
public static void main(String[] args) {
    input();
    // 1 번째 원소부터 M 번째 원소를 조건에 맞게 고르는 모든 방법을 탐색해줘
    rec_func( k: 1,  value: 0);
    // ans 가 정말 "진 부분집합"만 다루는 지 확인하기
    System.out.println(ans);
}
```

I 재귀 함수를 이용한 완전 탐색(Brute Force)

재귀 함수의 정의