

# 브루트 포스 - 재귀 (연습)

최백준 [choi@startlink.io](mailto:choi@startlink.io)

---

# 로또 2등차기

<https://www.acmicpc.net/problem/6603>

- 로또의 모든 조합을 출력해보는 문제

2

크기 가 있을 때

$6 \times \frac{12}{2} = 36$  (크는  $12^2$ )

$K \leq 12$

시작

O/X

$K \leq 9$

$2^{12}$

= 4096

1	2	4	8	15	16	23	42	43
0	x	0	0	0	0	0	0	x

<https://www.acmicpc.net/problem/6603>

• go(a, index, cnt)

- a: 입력으로 주어진 수
- index: 선택할지 말지 결정해야 하는 인덱스
- cnt: 현재까지 포함한 수의 개수

3은

③ 다음 경우 호출.

↳ 선택: go(a, index+1, cnt+1)  
 선택x: go(a, index+1, cnt)

0 ~ (k-1)

index

① 정답을 찾는 : cnt == 6

② 불가능한 경우

index == a.size()

# 로또

<https://www.acmicpc.net/problem/6603>

- go(a, index, cnt)
  - a: 입력으로 주어진 수
  - index: 선택할지 말지 결정해야 하는 인덱스
  - cnt: 현재까지 포함한 수의 개수
- 정답을 찾은 경우
  - cnt == 6
- 불가능한 경우
  - index == a.size()
- 다음 경우 (선택하는 것은 다른 배열을 사용)
  - index번째를 선택: go(a, index+1, cnt+1)
  - index번째를 선택하지 않음: go(a, index, cnt)

# 로또

<https://www.acmicpc.net/problem/6603>

- 소스: <http://codeplus.codes/4b77ac042e0342aebce47ffa395ae082>

# 부분수열의 합

<https://www.acmicpc.net/problem/1182>

- 서로 다른  $N$ 개의 정수로 이루어진 수열이 있을 때, 크기가 양수인 부분수열 중에서 그 수열의 원소를 다 더한 값이  $S$ 가 되는 경우의 수를 구하는 문제

- $1 \leq N \leq 20$

$$O(2^N)$$

$$2^N - 1$$

$$2^N$$

# 부분수열의 합

7

<https://www.acmicpc.net/problem/1182>

- go(index, sum)
  - index: 부분수열에 포함할지 말지 결정해야 하는 인덱스
  - sum: 현재까지 부분수열의 합

# 부분수열의 합

<https://www.acmicpc.net/problem/1182>

n개

0 ~ n-1



- go(index, sum)
  - index: 부분수열에 포함할지 말지 결정해야 하는 인덱스
  - sum: 현재까지 부분수열의 합

## 정답을 찾은 경우

- index == n && sum == r

## 불가능한 경우

- index == n && sum != r

## 다음 경우

- index번째를 부분수열에 추가: go(index+1, sum+a[i])
- index번째를 부분수열에 추가하지 않음: go(index+1, sum)



# 부분수열의 합

<https://www.acmicpc.net/problem/1182>

- 소스: <http://codeplus.codes/662b6e4b682e4d67bba4f01aa3b252c0>

# 부분수열의 합

<https://www.acmicpc.net/problem/14225>

- 수열  $S$ 가 주어졌을 때, 수열  $S$ 의 부분 수열의 합으로 나올 수 없는 가장 작은 자연수를 구하는 문제
- 예를 들어,  $S = [5, 1, 2]$ 인 경우에 1, 2, 3( $=1+2$ ), 5, 6( $=1+5$ ), 7( $=2+5$ ), 8( $=1+2+5$ )을 만들 수 있다. 하지만, 4는 만들 수 없기 때문에 정답은 4이다.

# 부분수열의 합

<https://www.acmicpc.net/problem/14225>

- S의 부분 수열의 개수는  $2^N$ 가지
- $N \leq 20$ 이기 때문에, 부분 수열을 모두 만들어 본다
- 부분 수열을 만드는 방법
  1. 재귀 호출
  2. 비트마스크

# 부분수열의 합

12

<https://www.acmicpc.net/problem/14225>

- 소스: <http://codeplus.codes/272fcb860f7840eb85e68385ef872675>

# 연산자 끼워넣기

<https://www.acmicpc.net/problem/14888>

- N개의 수로 이루어진 수열과 N-1개의 연산자가 있다. ( $2 \leq N \leq 11$ )
- 이 때, 수와 수 사이에 연산자를 하나씩 끼워넣어서 만들 수 있는 수식 결과의 최대값과 최소값을 구하는 문제
- 수식의 계산은 연산자 우선순위를 무시하고 앞에서부터 진행한다
- 수의 순서는 바꿀 수 없다

# 연산자 끼워넣기

14

<https://www.acmicpc.net/problem/14888>

• 수열 = [1, 2, 3, 4, 5, 6], 연산자 = +2개, -1개,  $\times$ 1개,  $\div$ 1개인 경우

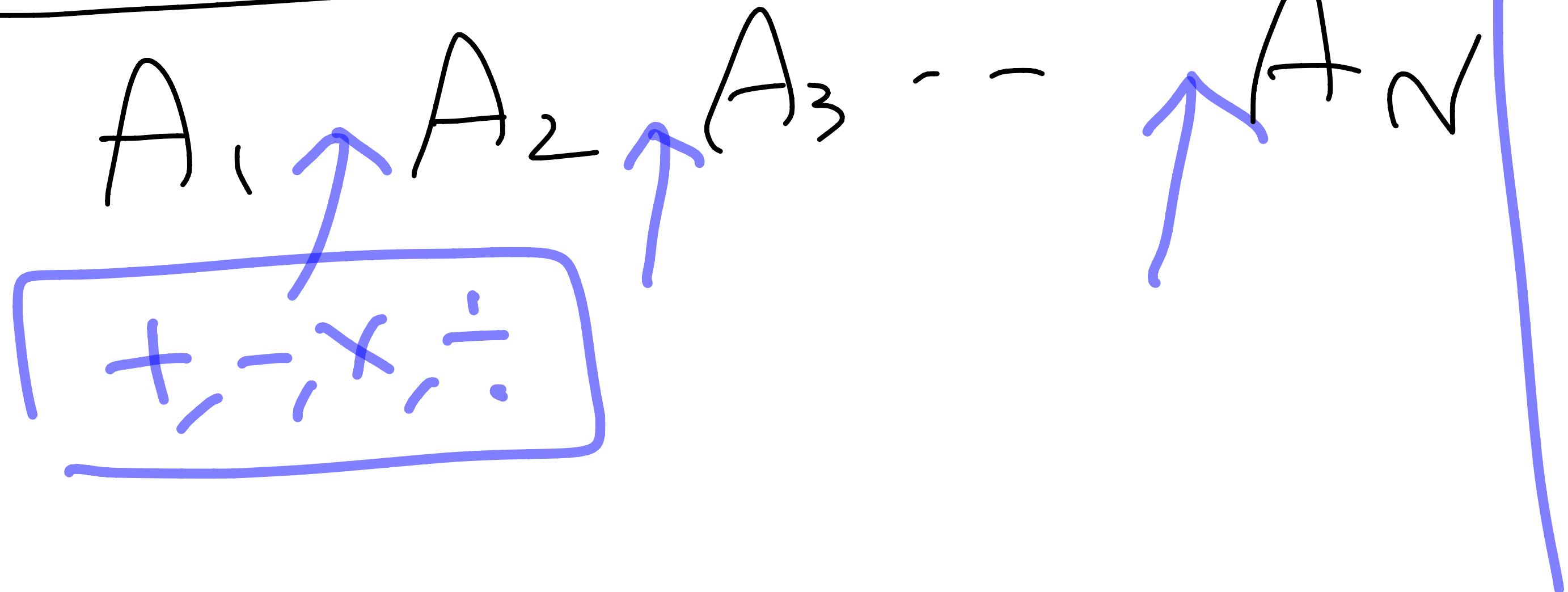
• 60가지가 가능하다

•  $1+2+3-4\times 5\div 6=1$

•  $1\div 2+3+4-5\times 6=12$

•  $1+2\div 3\times 4-5+6=5$

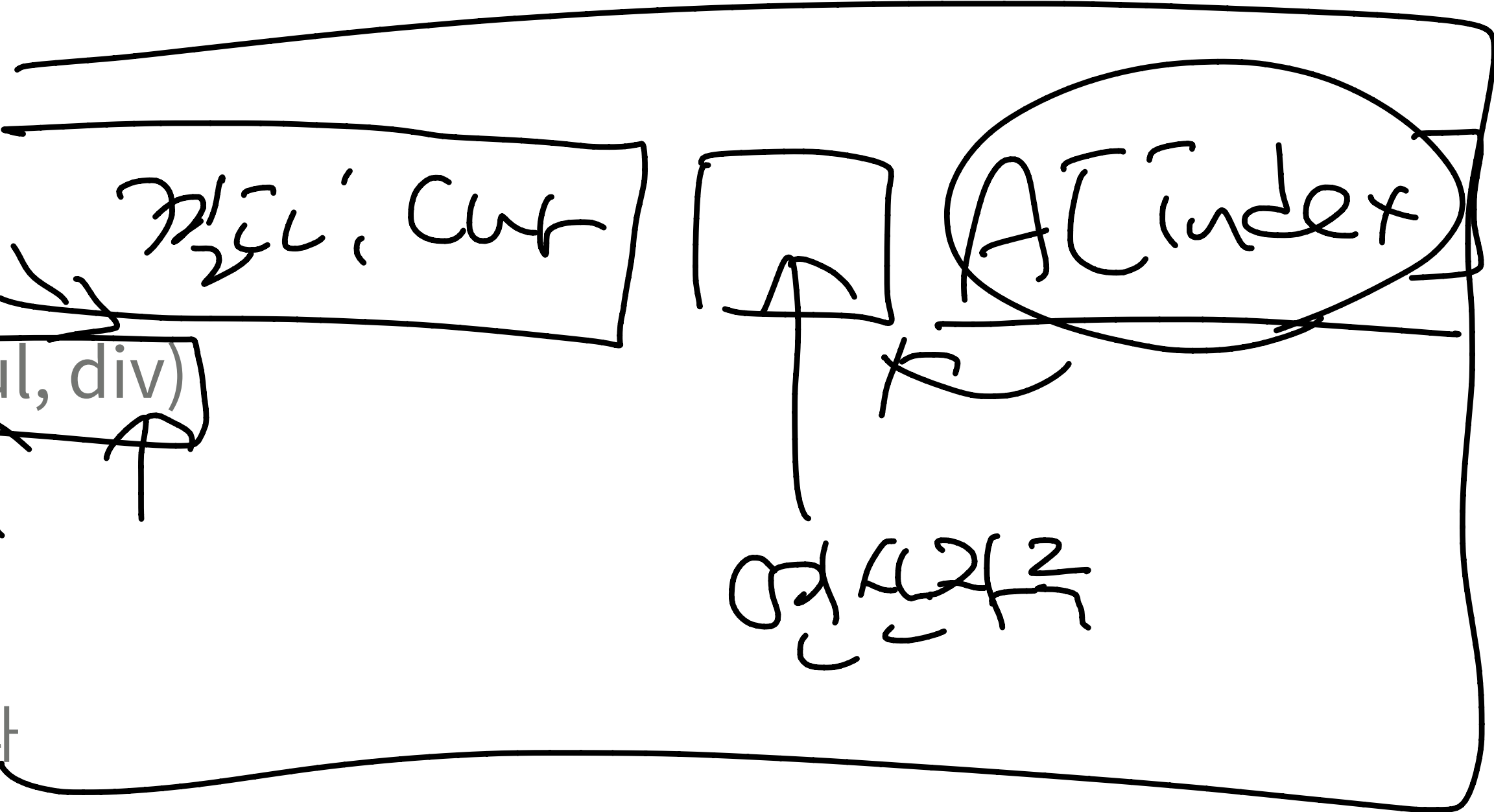
•  $1\div 2\times 3-4+5+6=7$



# 연산자 끼워넣기

<https://www.acmicpc.net/problem/14888>

- `go(a, index, cur, plus, minus, mul, div)`
  - a: 입력으로 주어진 수열
  - index: 현재 계산해야 하는 인덱스
  - cur: index-1번째까지 계산한 결과
  - plus, minus, mul, div: 사용할 수 있는 연산자의 개수



①  $+$ 를 사용: `go(a, index+1, cur + AT[index], plus-1, minus, mul, div);`

②  $-$ 를 사용: `go(a, index+1, cur - AT[index], plus, minus-1, mul, div);`

# 연산자 끼워넣기

<https://www.acmicpc.net/problem/14888>

- go(a, index, cur, plus, minus, mul, div)
  - a: 입력으로 주어진 수열
  - index: 현재 계산해야 하는 인덱스
  - cur: index-1번째까지 계산한 결과
  - plus, minus, mul, div: 사용할 수 있는 연산자의 개수

• 정답을 찾은 경우

- `index == n`

• 다음 경우

- + 사용: `go(a, index+1, cur+a[index], plus-1, minus, mul, div)`
- - 사용: `go(a, index+1, cur-a[index], plus, minus-1, mul, div)`
- $\times$  사용: `go(a, index+1, cur*a[index], plus, minus, mul-1, div)`
- / 사용: `go(a, index+1, cur/a[index], plus, minus, mul, div-1)`



# 연산자 끼워넣기

<https://www.acmicpc.net/problem/14888>

- 소스: <http://codeplus.codes/14e87f7cbc9f466694f5be80c458f230>

# 연산자 끼워넣기 (2)

<https://www.acmicpc.net/problem/15658>

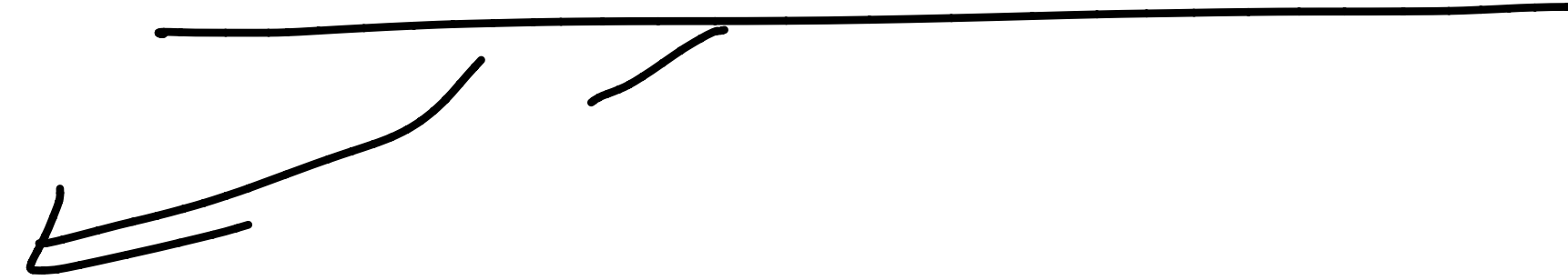
- N개의 수로 이루어진 수열과 **N-1개 이상**의 연산자가 있다. ( $2 \leq N \leq 11$ )
- 이 때, 수와 수 사이에 연산자를 하나씩 끼워넣어서 만들 수 있는 수식 결과의 최대값과 최소값을 구하는 문제
- 수식의 계산은 연산자 우선순위를 무시하고 앞에서부터 진행한다
- 수의 순서는 바꿀 수 없다

# 연산자 끼워넣기 (2)

<https://www.acmicpc.net/problem/15658>

- 수열 = [1, 2, 3, 4, 5, 6], 연산자 = +3개, -2개,  $\times$  1개,  $\div$  1개인 경우

- 250까지 가능하다



- $1+2+3-4\times 5\div 6 = 1$
- $1\div 2+3+4-5\times 6 = 12$
- $1+2\div 3\times 4-5+6 = 5$
- $1\div 2\times 3-4+5+6 = 7$
- $1+2+3+4-5-6 = -1$
- $1+2+3-4-5\times 6 = -18$

# 연산자 끼워넣기 (2)

<https://www.acmicpc.net/problem/15658>

- go(a, index, cur, plus, minus, mul, div)
  - a: 입력으로 주어진 수열
  - index: 현재 계산해야 하는 인덱스
  - cur: index-1번째까지 계산한 결과
  - plus, minus, mul, div: 사용할 수 있는 연산자의 개수

# 연산자 끼워넣기 (2)

<https://www.acmicpc.net/problem/15658>

- `go(a, index, cur, plus, minus, mul, div)`
  - `a`: 입력으로 주어진 수열
  - `index`: 현재 계산해야 하는 인덱스
  - `cur`: `index-1`번째까지 계산한 결과
  - `plus, minus, mul, div`: 사용할 수 있는 연산자의 개수
- 정답을 찾은 경우
  - `index == n`
- 다음 경우
  - `+` 사용: `go(a, index+1, cur+a[index], plus-1, minus, mul, div)`
  - `-` 사용: `go(a, index+1, cur-a[index], plus, minus-1, mul, div)`
  - `×` 사용: `go(a, index+1, cur*a[index], plus, minus, mul-1, div)`
  - `/` 사용: `go(a, index+1, cur/a[index], plus, minus, mul, div-1)`

# 연산자 끼워넣기 (2)

<https://www.acmicpc.net/problem/15658>

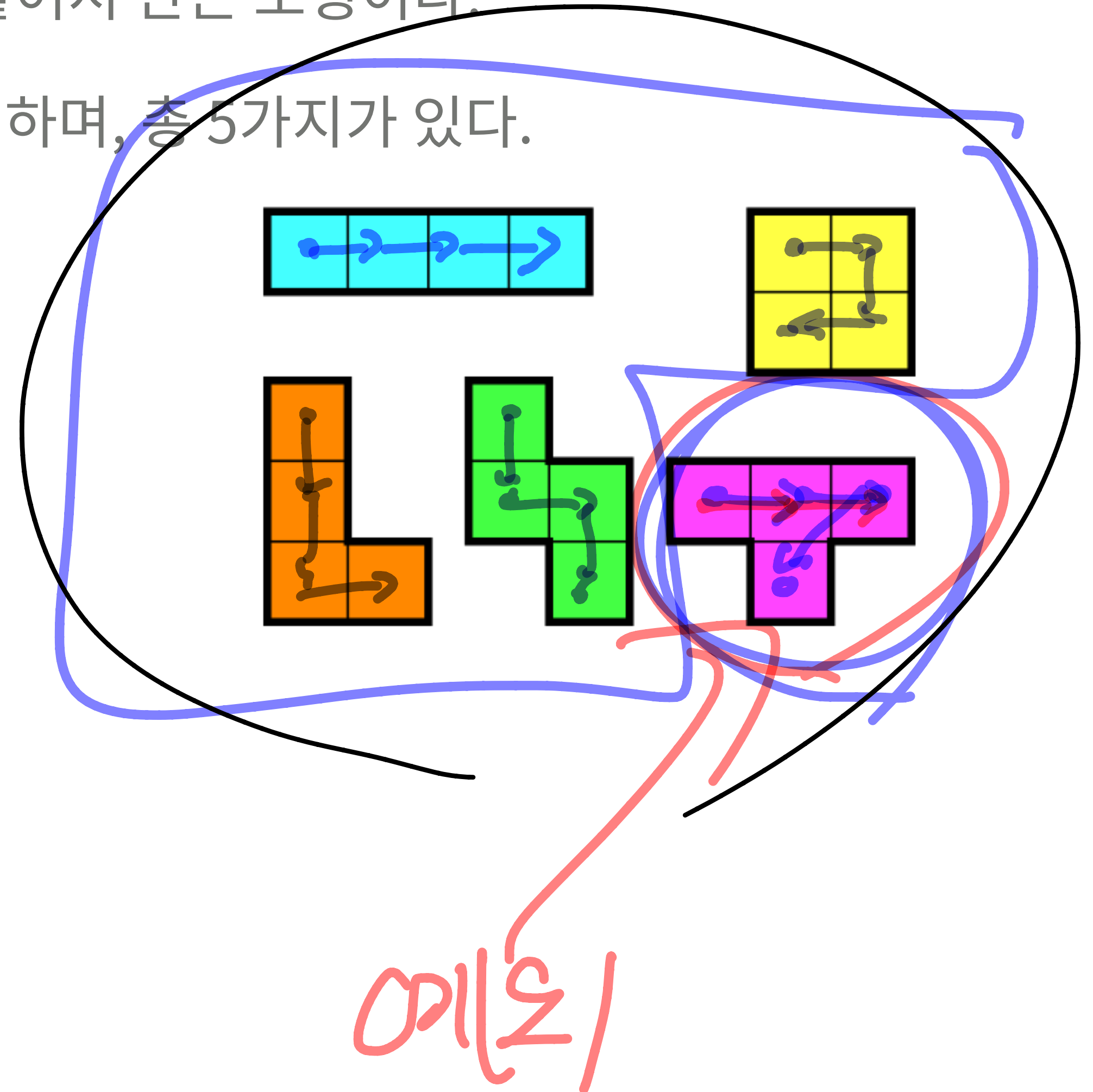
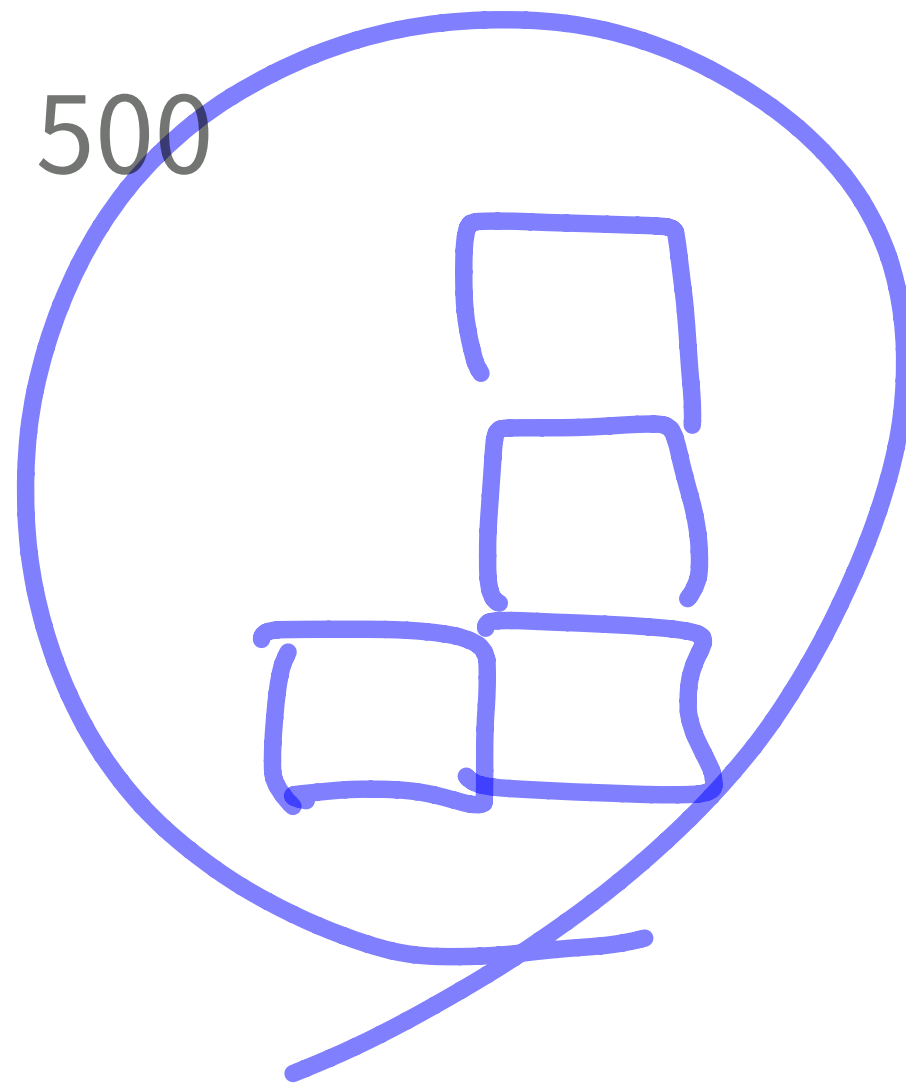
- 소스: <http://codeplus.codes/05f954738eda49fabb398ad7eb422465>
- 연산자 끼워넣기와 같은 소스로 해결할 수 있다

# 테트로미노

<https://www.acmicpc.net/problem/14500>

개3)

- 폴리오미노는 크기가  $1 \times 1$ 인 정사각형을 여러 개 이어 붙여서 만든 도형이다.
- 정사각형 4개를 이어 붙인 폴리오미노는 테트로미노라고 하며, 총 5가지가 있다.
- $N \times M$  크기의 종이 위에 테트로미노를 하나 놓아서
- 놓인 칸에 쓰여 있는 수의 합을 최대로 하는 문제
- $4 \leq N, M \leq 500$

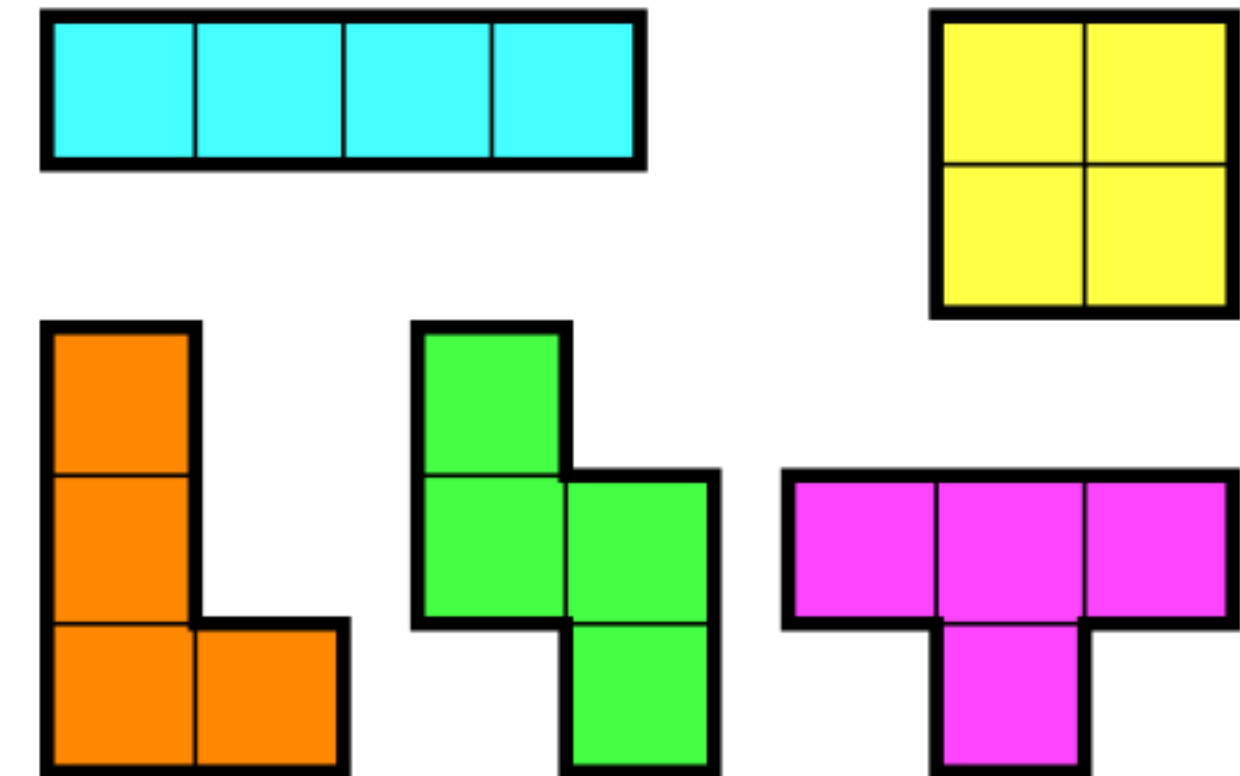


예2)

# 테트로미노

<https://www.acmicpc.net/problem/14500>

- 테트로미노는 총 19가지가 있고
- 하나의 테트로미노당 놓을 수 있는 방법의 개수는 약,  $O(NM)$ 가지 이다
- 경우의 수가 많지 않기 때문에
- 각각의 테트로미노에 대해서 모든 칸에 놓아본다

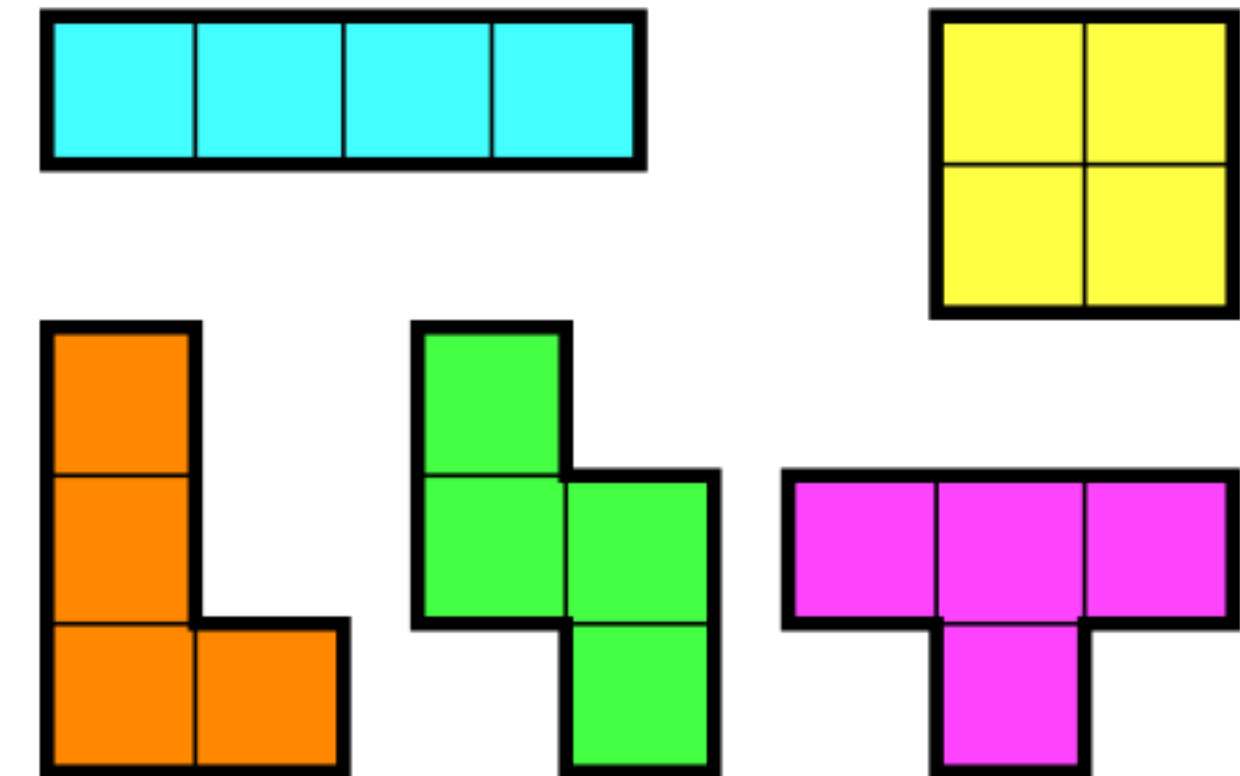




# 테트로미노

<https://www.acmicpc.net/problem/14500>

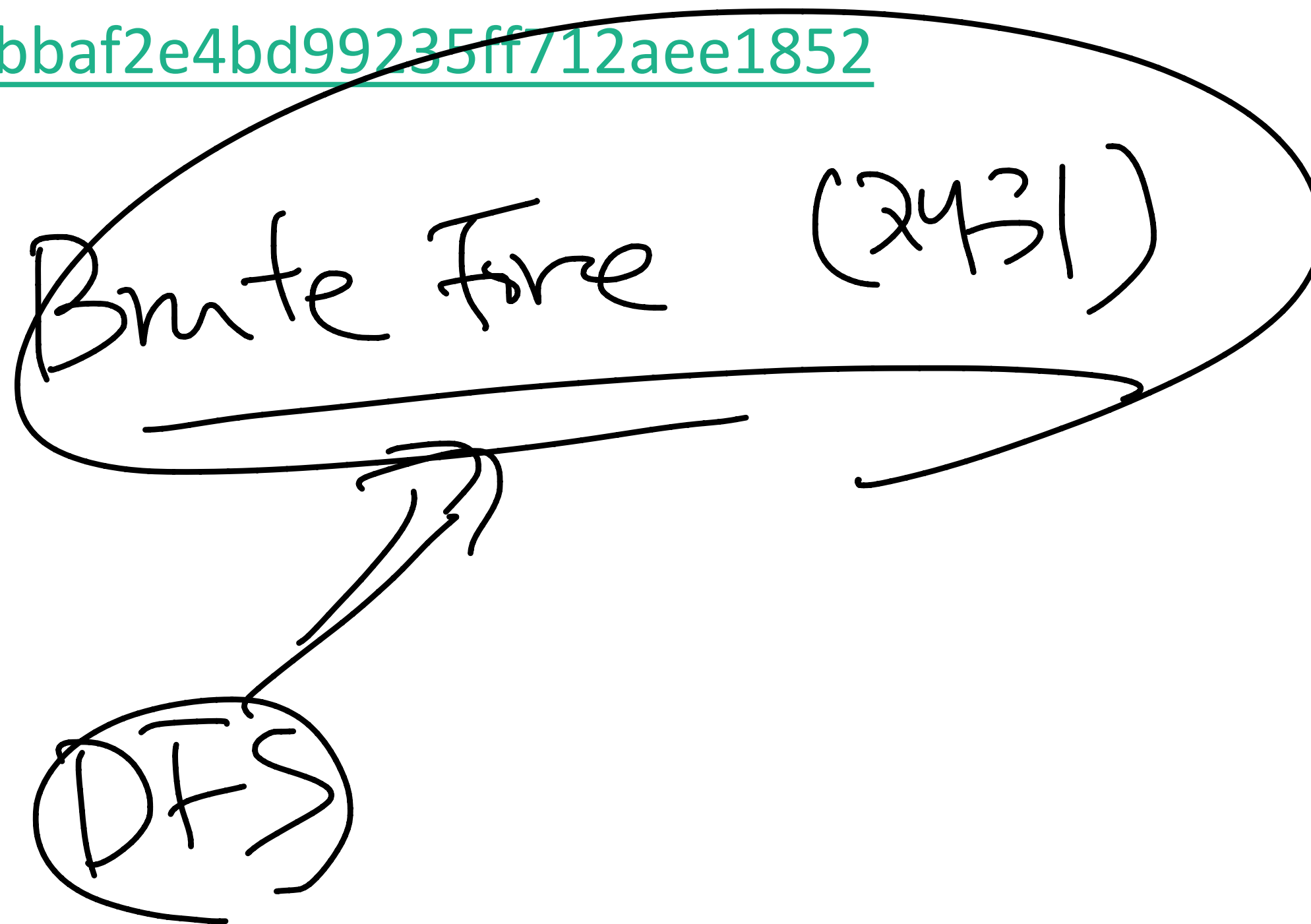
- 하나를 제외한 나머지 테트로미노는 임의의 한 칸에서 시작해서
- 3개의 칸을 연속해서 방문한 형태이다.
- 하나는 재귀 함수로는 할 수 없기 때문에
- for문으로 살펴본다.



# 테트로미노

<https://www.acmicpc.net/problem/14500>

- 소스: <http://codeplus.codes/4c5403bbaf2e4bd99235ff712aee1852>



# 두 동전

$N \times M$

<https://www.acmicpc.net/problem/16197>

- $N \times M$  크기의 보드에 4개의 버튼이 있다.
- 칸은 비어있거나, 동전 벽이다.
- 동전은 2개이다.
- 버튼은 왼쪽, 오른쪽, 위, 아래이고, 누르면 그 방향으로 이동한다.
- 이동하려는 칸이 벽이면 이동하지 않는다.
- 이동하려는 칸이 없으면 보드 바깥으로 떨어진다.
- 그 외에는 이동한다.
- 두 동전 중 하나만 보드에 떨어뜨리기 위해 버튼을 몇 번 눌러야 하는가?
- 10번보다 많이 눌러야 하면 -1을 출력한다.

$$4^{10} = 2^{20} = 1048576$$

최소값

# 두 동전

<https://www.acmicpc.net/problem/16197>

- 총 4개의 방향을 10번까지 수행할 수 있다.
- 방법의 수:  $4^{10}$

# 두 동전

<https://www.acmicpc.net/problem/16197>

• go(step, x1, y1, x2, y2)

• step: 버튼을 누른 횟수

• (x1, y1): 한 동전의 위치

• (x2, y2): 다른 동전의 위치

(x1, y1)

(x2, y2)

① 버튼을 누른 경우

Step += 1

두 동전 모두

떨어진 경우

② 버튼을 누르지 않는 경우

두 동전 중 하나

떨어지지 않는 경우

③ 이동 방향

→ ← ↑ ↓

# 두 동전

<https://www.acmicpc.net/problem/16197>

- `go(step, x1, y1, x2, y2)`
  - `step`: 버튼을 누른 횟수
  - `(x1, y1)`: 한 동전의 위치
  - `(x2, y2)`: 다른 동전의 위치
- 불가능한 경우
  - `step == 11`
  - 동전이 둘 다 떨어진 경우
- 정답을 찾은 경우
  - 동전 하나만 떨어진 경우
- 다음 경우
  - `go(step+1, nx1, ny1, nx2, ny2)`

# 두 동전

<https://www.acmicpc.net/problem/16197>

- 소스: <http://codeplus.codes/f60260f4948b41e59811e7c0c8ece274>

# 에너지 모으기

<https://www.acmicpc.net/problem/16198>

• N개의 에너지 구슬이 있고, i번째 에너지 구슬의 무게는  $W[1], W[2], \dots, W[N]$  이다.  $N \leq 10$

• 에너지를 모으는 방법은 다음과 같다.

1. 에너지 구슬을 하나 고른다. 첫 번째와 마지막은 고를 수 없다. 고른 구슬의 번호를 x라고 한다.

2. x번째 구슬을 제거한다.

3.  $W[x-1] \times W[x+1]$ 의 에너지를 모은다.

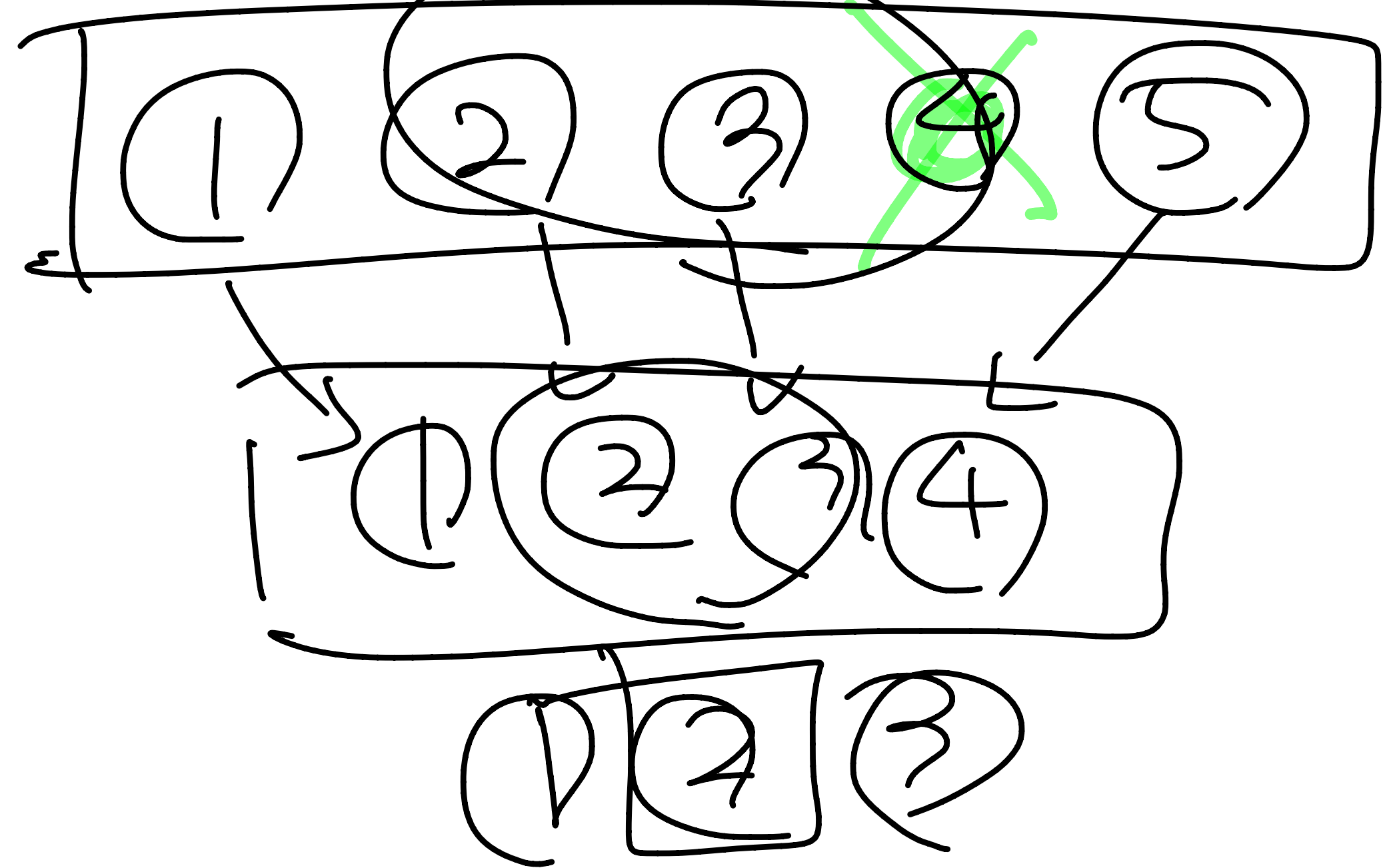
4. N을 1 감소시키고, 구슬의 번호를 다시 매긴다.

$$\begin{array}{lcl} N & \rightarrow & (N-2) \\ N-1 & \rightarrow & (N-3) \\ & \vdots & \\ 3 & \rightarrow & 1 \end{array}$$

③

계승리식:  $(N-2)!$

$$8! = 4 \times 3 \times 2 \times 1$$





# 에너지 모으기

<https://www.acmicpc.net/problem/16198>

- $N$ 이 10보다 작거나 같기 때문에, 합칠 수 있는 방법의 수가 크지 않다.

# 에너지 모으기

<https://www.acmicpc.net/problem/16198>

- $go(w)$ : 에너지 구슬의 무게가  $w$ 에 순서대로 저장되어 있을 때, 모을 수 있는 에너지의 최댓값

# 에너지 모으기

<https://www.acmicpc.net/problem/16198>

- 소스: <http://codeplus.codes/21aeddefc2264469ae39a09ce973d152>

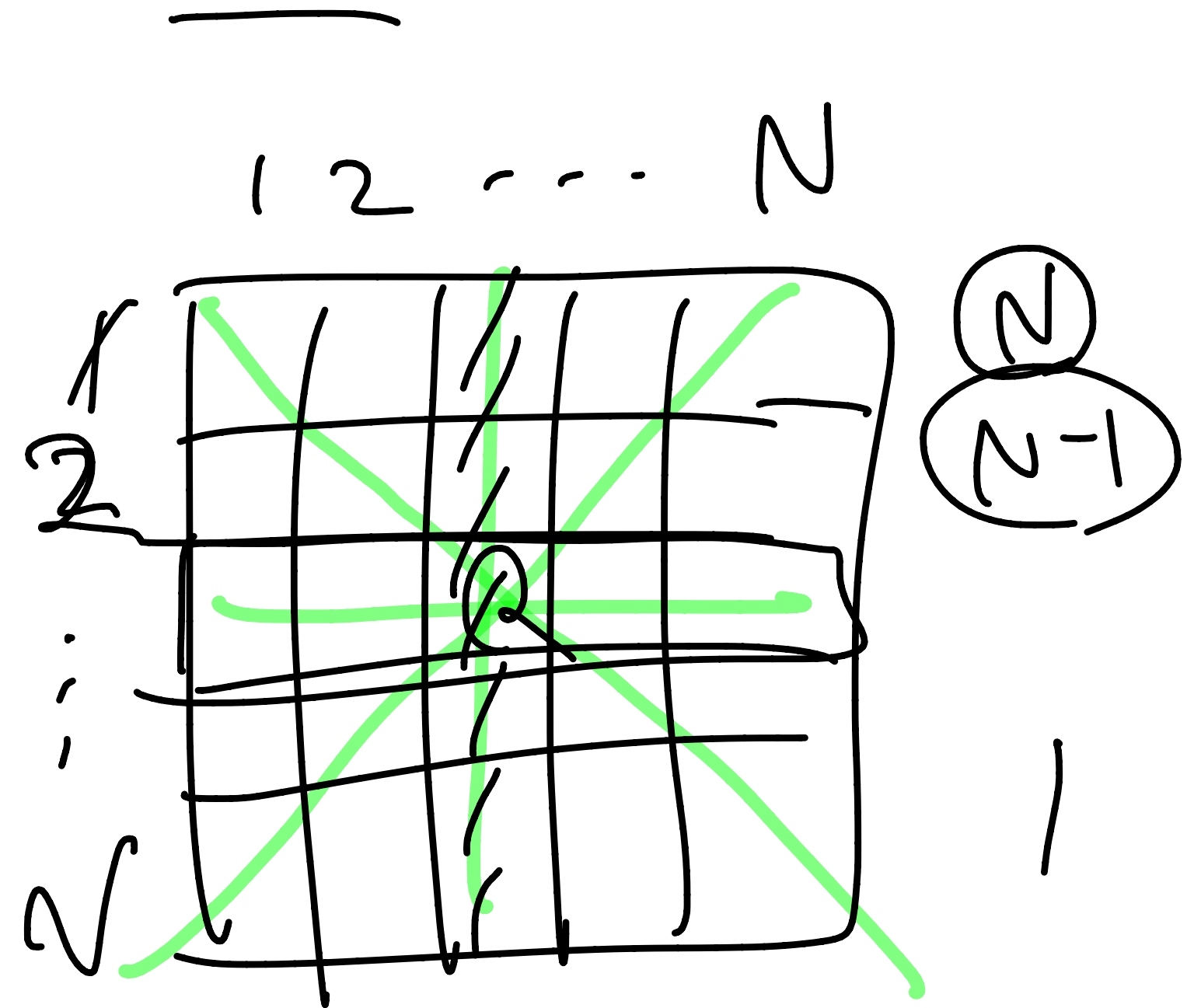
# 백트래킹

---

# N-Queen

<https://www.acmicpc.net/problem/9663>

- $N \times N$  크기의 체스판 위에 Queen을 N개 놓는 방법의 수를 구하는 문제



각이 퀸이  $\frac{N!}{N^2}$   $\frac{N!}{N^2}$   
 $N^2$

$$\begin{aligned} 2^{N^2} &= 2^{N \times N} \\ \textcircled{N^2} &\rightarrow \textcircled{N!} \end{aligned}$$

# N-Queen

<https://www.acmicpc.net/problem/9663>

- calc(row): row 행에 퀸을 어디에 놓을지 결정해야 함

row를

컬을

# N-Queen

<https://www.acmicpc.net/problem/9663>

39

$0 \sim (n-1)$

- calc(row): row 행에 퀸을 어디에 놓을지 결정해야 함

```
void calc(int row) {
```

```
    if (row == n) {
```

```
        ans += 1;
```

```
    }
```

```
    for (int col=0; col<n; col++) {
```

```
        a[row][col] = true;
```

```
        if (check(row, col)) {
```

```
            calc(row+1);
```

```
        }
```

```
        a[row][col] = false;
```

```
    }
```

```
}
```

$0 \sim n$

row, col

# N-Queen

40

<https://www.acmicpc.net/problem/9663>

- 소스: <http://codeplus.codes/d162cdcf9a24479ca68ace4c1e53c4f5>



# N-Queen

<https://www.acmicpc.net/problem/9663>

- check\_col[i] = i번 열에 퀸이 놓여져 있으면 true

false

	0	1	2	3	4	5
0	0	1	2	3	4	5
1	0	1	2	3	4	5
2	0	1	2	3	4	5
3	0	1	2	3	4	5
4	0	1	2	3	4	5
5	0	1	2	3	4	5

# N-Queen

42

<https://www.acmicpc.net/problem/9663>

- `check_dig[i] = /` 대각선에 퀸이 있으면

*true*

*row + col*

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

# N-Queen

<https://www.acmicpc.net/problem/9663>

43

row-col + (n-1)

- `check_dig2[i] = \` 대각선에 퀸이 있으면

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

# N-Queen

<https://www.acmicpc.net/problem/9663>

- Check 부분을 배열을 이용하면 놓을 수 있는지 검사를  $O(1)$ 만에 해결 할 수 있다.
- 소스: <http://codeplus.codes/1bf652538a684852aa8f45c48df01cfd>

# 스도쿠

<https://www.acmicpc.net/problem/2580>

- 스도쿠를 푸는 문제

크게 어렵지!

Brute Force

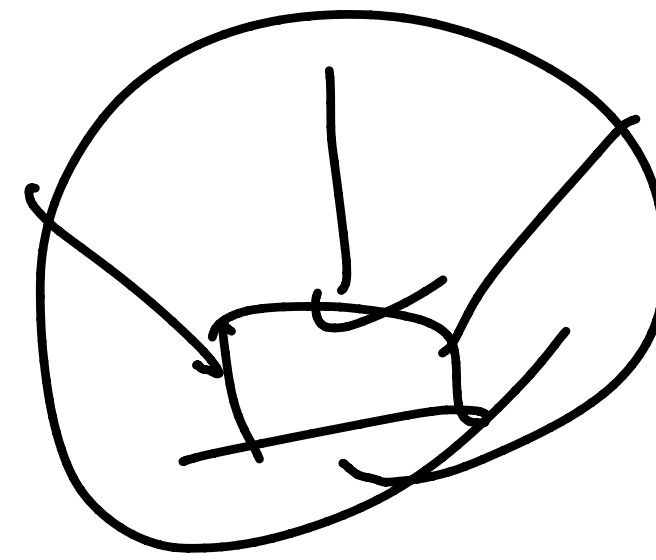
45

<del>2</del>	3	5	4	6	9	2	7	8
7	8	2	1		5	6		9
	6		2	7	8	1	3	5
3	2	1		4	6	8	9	7
8		4	9	1	3	5		6
5	9	6	8	2		4	1	3
9	1	7	6	5	2		8	
6		3	7		1	9	5	2
2	5	8	3	9	4	7	6	

1	3	5	4	6	9	2	7	8
7	8	2	1	3	5	6	4	9
4	6	9	2	7	8	1	3	5
3	2	1	5	4	6	8	9	7
8	7	4	9	1	3	5	2	6
5	9	6	8	2	7	4	1	3
9	1	7	6	5	2	3	8	4
6	4	3	7	8	1	9	5	2
2	5	8	3	9	4	7	6	1

# 스도쿠

9x9



46

<https://www.acmicpc.net/problem/2580>

- go(z): z번째 칸을 채우는 함수
- (x, y)  $\rightarrow 9 \times x + y$  번째 칸

(x항, y항)

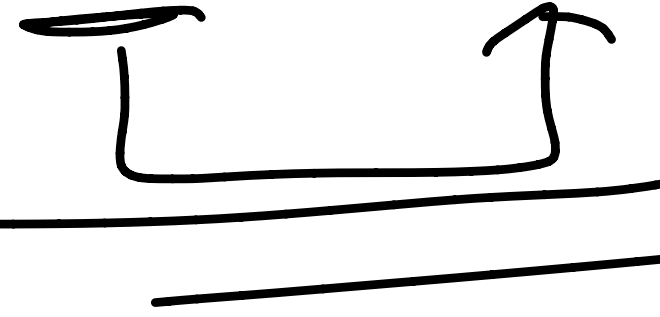
$9x + y$  번째 칸

0	1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16	17
18	19	20	21	22	23	24	25	26
27	28	29	30	31	32	33	34	35
36	37	38	39	40	41	42	43	44
45	46	47	48	49	50	51	52	53
54	55	56	57	58	59	60	61	62
63	64	65	66	67	68	69	70	71
72	73	74	75	76	77	78	79	80

# 스도쿠

<https://www.acmicpc.net/problem/2580>

- $c[i][j] = i$ 행에 숫자  $j$ 가 있으면 true



0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8

- $c2[i][j]$  = i열에 숫자 j가 있으면 true

[illegible]



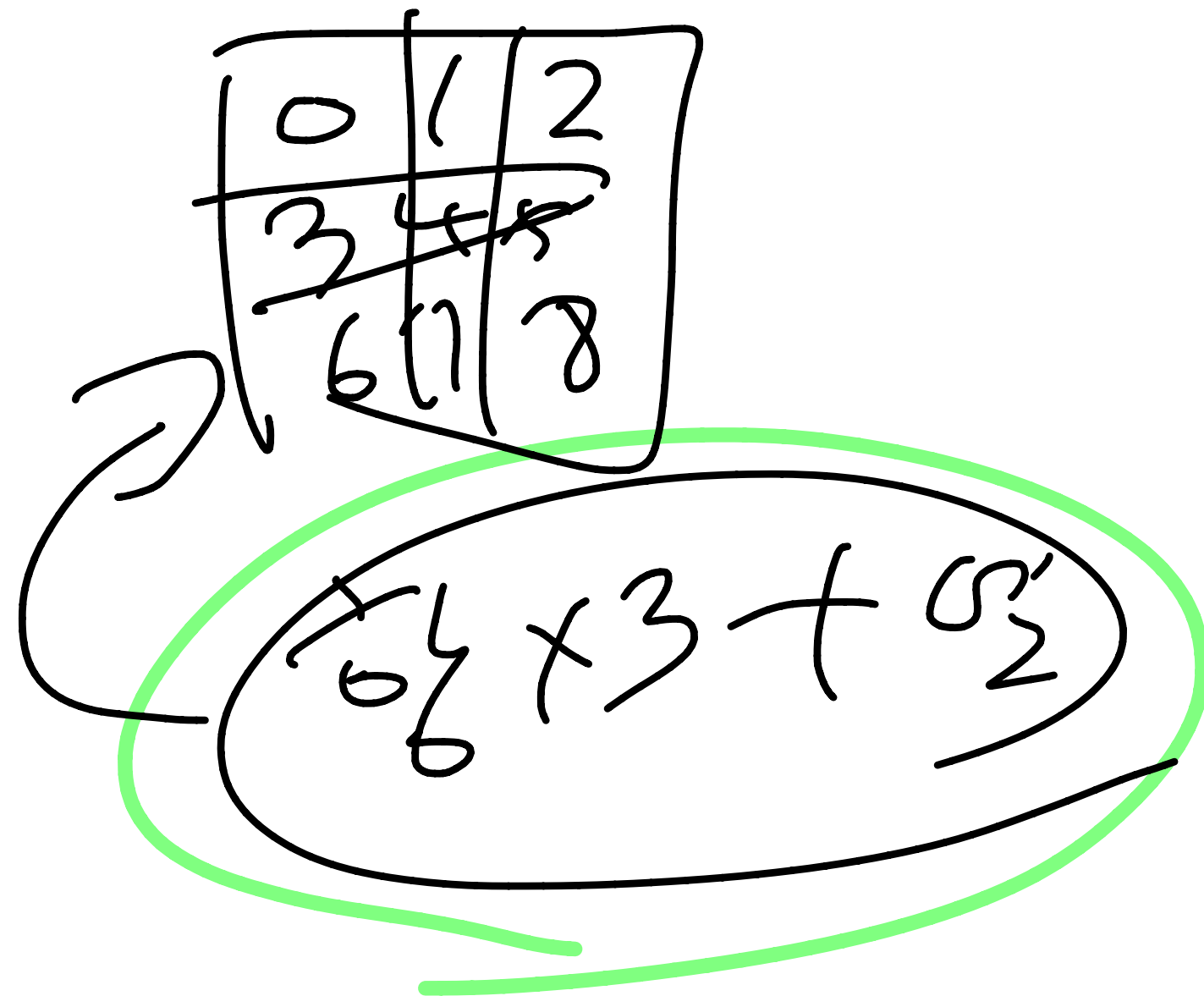
# 스도쿠

49

<https://www.acmicpc.net/problem/2580>

•  $c3[i][j] = i$ 번 작은 정사각형에 숫자  $j$ 가 있으면 true

•  $(x, y)$ 는  $(x/3)*3 + (y/3)$ 번째 칸



0 1 2

0

1

2

0	0	0	1	1	1	2	2	2
0	0	0	1	1	1	2	2	2
0	0	0	1	1	1	2	2	2
3	3	3	4	4	4	5	5	5
3	3	3	4	4	4	5	5	5
3	3	3	4	4	4	5	5	5
6	6	6	7	7	7	8	8	8
6	6	6	7	7	7	8	8	8
6	6	6	7	7	7	8	8	8

2

# 스도쿠

50

<https://www.acmicpc.net/problem/2580>

```
for (int i=0; i<n; i++) {  
    for (int j=0; j<n; j++) {  
        cin >> a[i][j];  
        if (a[i][j] != 0) {  
            c[i][a[i][j]] = true;  
            c2[j][a[i][j]] = true;  
            c3[square(i,j)][a[i][j]] = true;  
        }  
    }  
}  
go(0);
```

Handwritten notes: *정답이 0이 아니다* (The answer is not 0) with an arrow pointing to the `if (a[i][j] != 0)` condition. The entire code block is circled, and the `go(0);` line is also circled.

# 스도쿠

<https://www.acmicpc.net/problem/2580>

```
void go(int z) {  
    if (z == 81) {  
        // check  
        exit(0);  
    }
```

```
}
```

```
int x = z/n, y = z%n;
```

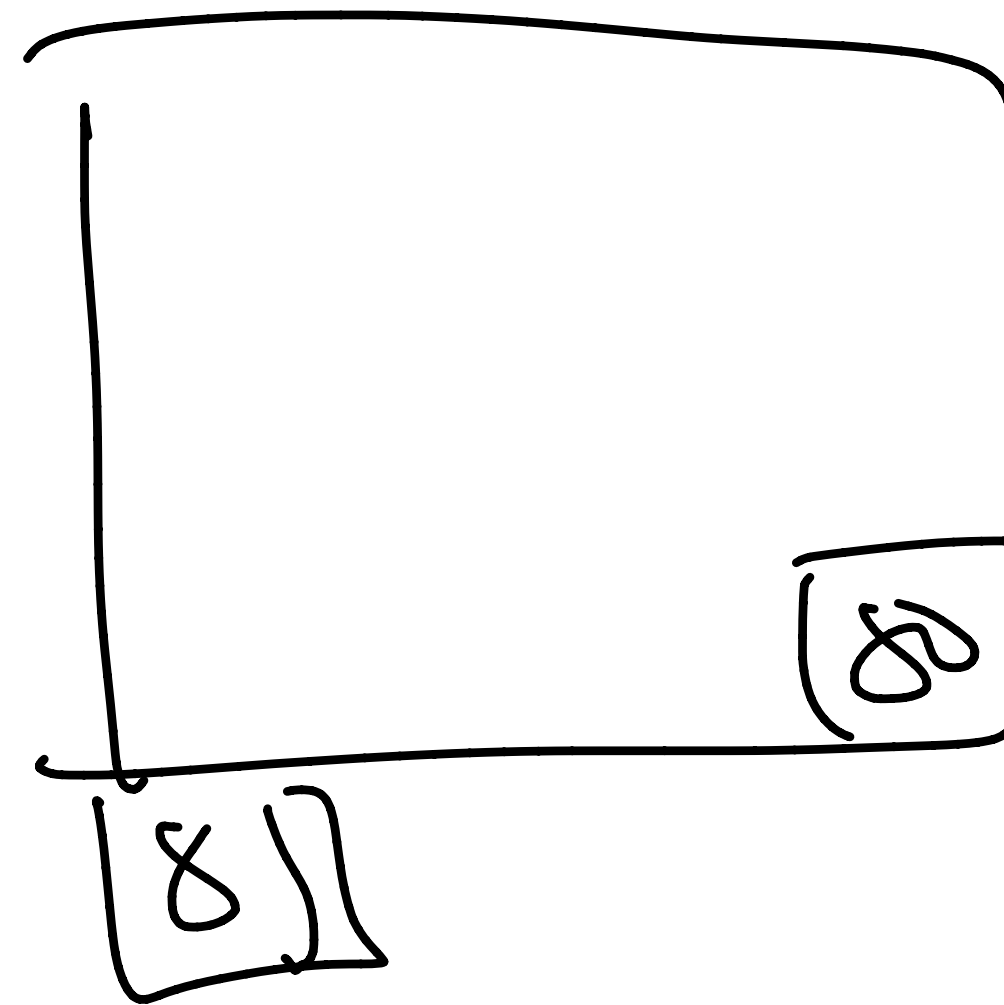
```
if (a[x][y] != 0) {  
    go(z+1);  
}
```

```
else {
```

```
    // next
```

```
}
```

```
}
```



72 0/0/2

<https://www.acmicpc.net/problem/2580>

```
// check
for (int i=0; i<n; i++) {
    for (int j=0; j<n; j++) {
        cout << a[i][j] << ' ';
    }
    cout << '\n';
}
exit(0);
```

$x^2$      $y^2$

<https://www.acmicpc.net/problem/2580>

```
// next
```

```

for (int i=1; i<9; i++) {
    if (c[x][i] == 0 && c2[y][i] == 0 && c3[square(x,y)][i] == 0) {
        c[x][i] = c2[y][i] = c3[square(x,y)][i] = true;
        a[x][y] = i;
        go(z+1);
        a[x][y] = 0;
        c[x][i] = c2[y][i] = c3[square(x,y)][i] = false;
    }
}

```

# 스도쿠

<https://www.acmicpc.net/problem/2580>

- 소스: <http://codeplus.codes/e71328fac9884189a7ffcc9bf0d704f7>

# 스도미노쿠

<https://www.acmicpc.net/problem/4574>

- 스도미노쿠를 푸는 문제

	1	2	3	4	5	6	7	8	9
A		7	2						5
B		6	1				8	4	2
C			9	2	1	8			7
D			6						3
E	7		8		6				
F	4				3	2	7	6	
G				5	9				
H									
I	9			4				7	8

	1	2	3	4	5	6	7	8	9
A	8	7	2	6	4	3	1	9	5
B	3	6	1	9	7	5	8	4	2
C	5	4	9	2	1	8	6	3	7
D	1	2	6	7	5	4	9	8	3
E	7	3	8	1	6	9	2	5	4
F	4	9	5	8	3	2	7	6	1
G	2	8	4	5	9	7	3	1	6
H	6	5	7	3	8	1	4	2	9
I	9	1	3	4	2	6	5	7	8

# 스도미노쿠

<https://www.acmicpc.net/problem/4574>

- 첫 번째 칸부터 수를 하나씩 차례대로 채워본다.



# 스도미노쿠

<https://www.acmicpc.net/problem/4574>

- 소스: <http://codeplus.codes/733f74f8ee774fd3a5fdadf2096aa362>

domino

# 알파벳

<https://www.acmicpc.net/problem/1987>

- 세로 R칸, 가로 C칸으로 된 표 모양의 보드가 있다
- 보드의 각 칸에는 대문자 알파벳이 하나씩 적혀 있고, 좌측 상단 칸 (1행 1열) 에는 말이 놓여 있다
- 말은 상하좌우로 인접한 네 칸 중의 한 칸으로 이동할 수 있다
- 같은 알파벳이 적힌 칸을 두 번 지날 수 없다
- 좌측 상단에서 시작해서, 말이 최대한 몇 칸을 지날 수 있는지를 구하는 문제

# 알파벳

<https://www.acmicpc.net/problem/1987>

- go(board, check, x, y, cnt)
  - board: 보드
  - check: 방문한 알파벳
  - x, y: 현재 위치
  - cnt: 방문한 칸의 수

# 알파벳

<https://www.acmicpc.net/problem/1987>

- go(board, check, x, y, cnt)
  - board: 보드
  - check: 방문한 알파벳
  - x, y: 현재 위치
  - cnt: 방문한 칸의 수
- 새로운 칸 nx, ny로 이동할 수 있는 경우
  - go(board, check, nx, ny, cnt+1)
    - 이 때, check는 변경해 줘야함

# 알파벳

<https://www.acmicpc.net/problem/1987>

61

0 1 2 3 4 5 6 7 8 9  
2  
↓

```
void go(vector<string> &board, vector<bool> &check, int x, int y, int cnt) {
```

```
    if (cnt > ans) ans = cnt;
```

```
    for (int k=0; k<4; k++) {
```

```
        int nx = x+dx[k];
```

```
        int ny = y+dy[k];
```

```
        if (nx >= 0 && nx < board.size() && ny >= 0 && ny < board[0].size()) {
```

```
            if (check[board[nx][ny]-'A'] == false) {
```

```
                check[board[nx][ny]-'A'] = true;
```

```
                go(board, check, nx, ny, cnt+1);
```

```
                check[board[nx][ny]-'A'] = false;
```

```
            }
```

```
        }
```

```
    }
```

```
}
```

nx, ny

x, y

# 알파벳

<https://www.acmicpc.net/problem/1987>

- go(board, check, x, y)
- board: 보드
- check: 방문한 알파벳
- x, y: 현재 위치
- 리턴 값: 방문할 수 있는 칸의 최대 개수
- 의미: (x, y)에서 이동을 시작하고, 방문한 알파벳이 check일 때, 방문할 수 있는 칸의 최대 개수

# 알파벳

<https://www.acmicpc.net/problem/1987>

```
int go(vector<string> &board, vector<bool> &check, int x, int y) {
    int ans = 0;
    for (int k=0; k<4; k++) {
        int nx = x+dx[k], ny = y+dy[k];
        if (nx >= 0 && nx < board.size() && ny >= 0 && ny <
board[0].size()) {
            if (check[board[nx][ny]-'A'] == false) {
                check[board[nx][ny]-'A'] = true;
                int next = go(board, check, nx, ny);
                if (ans < next) ans = next;
                check[board[nx][ny]-'A'] = false;
            }
        }
    }
    return ans + 1;
}
```

# 알파벳

<https://www.acmicpc.net/problem/1987>

- 소스: <http://codeplus.codes/d86cd86c235546c2a77abe1654213dde>