

다이나믹 프로그래밍 4

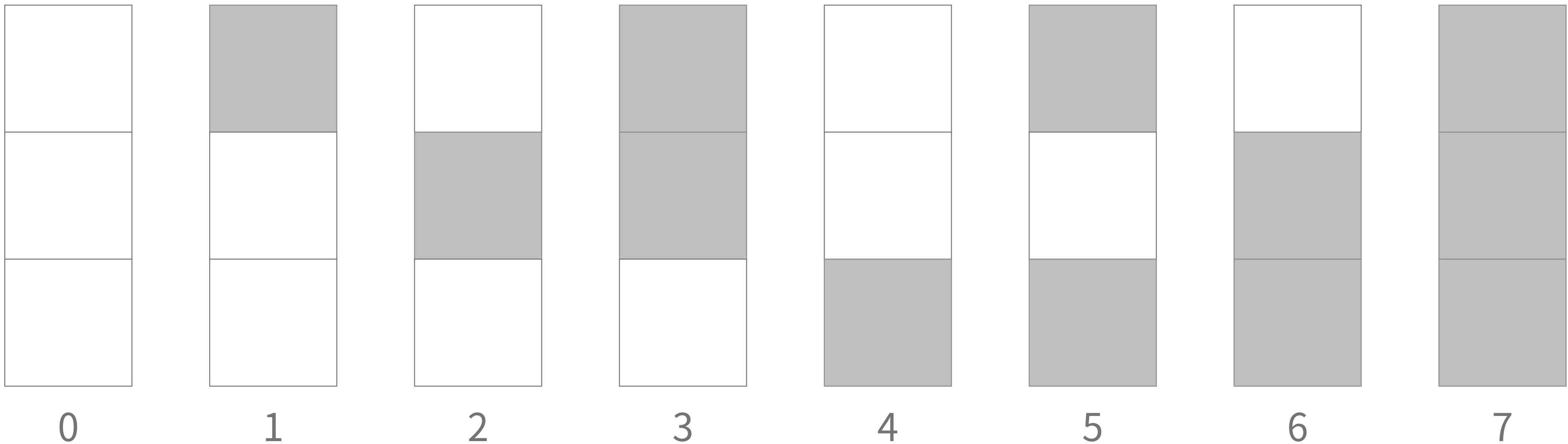
최백준 choi@startlink.io

상태 다이나믹

타일 채우기

<https://www.acmicpc.net/problem/2133>

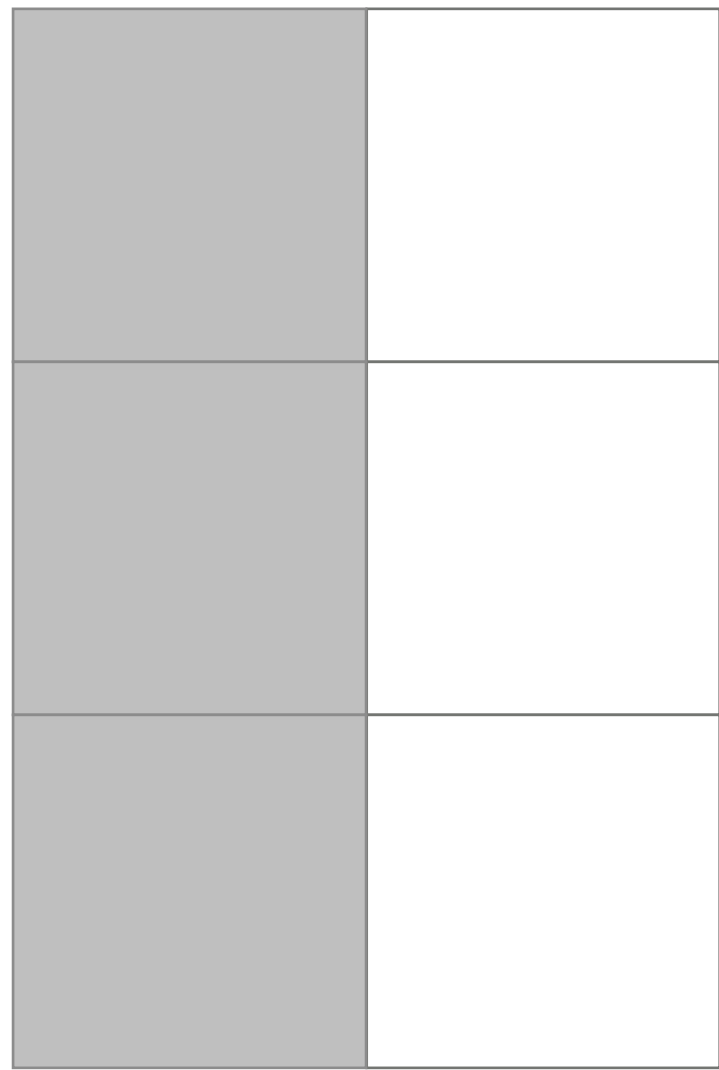
- $3 \times N$ 을 1×2 , 2×1 로 채우는 방법의 수
- $D[i][j] = 3 \times i$ 를 채우는 방법의 수, i 열의 상태는 j
- 마지막에 올 수 있는 가능한 경우의 수 (회색: 채워져 있는 칸)



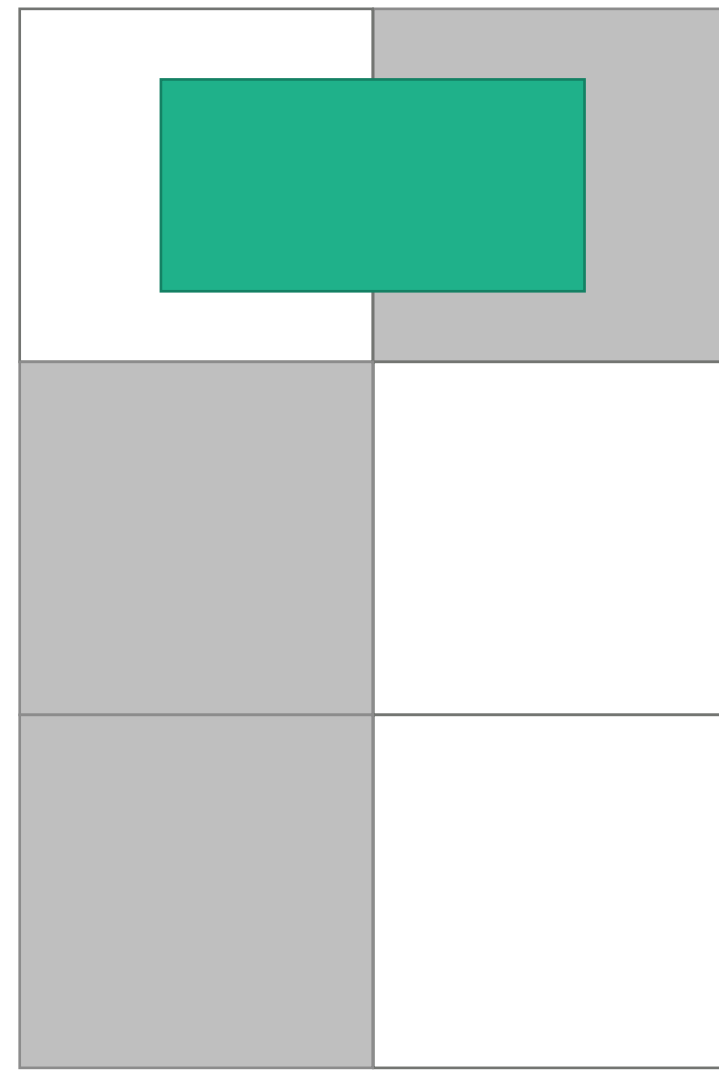
타일 채우기

<https://www.acmicpc.net/problem/2133>

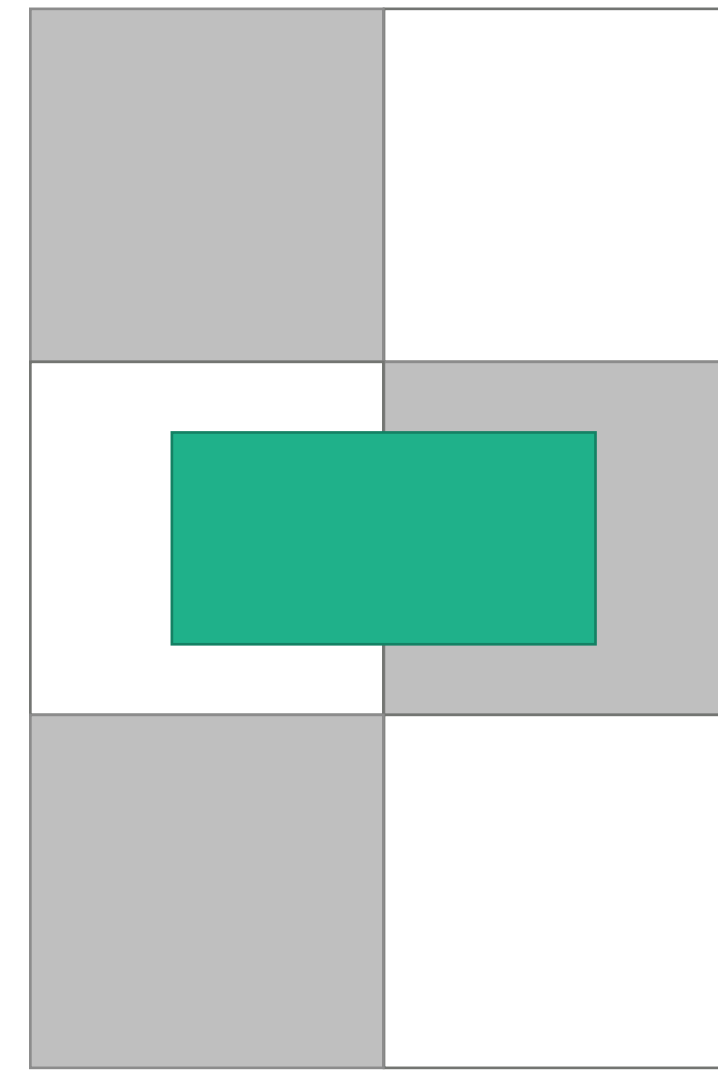
- i 열을 채울 때, $i-1$ 에 빈 칸이 있으면 안된다



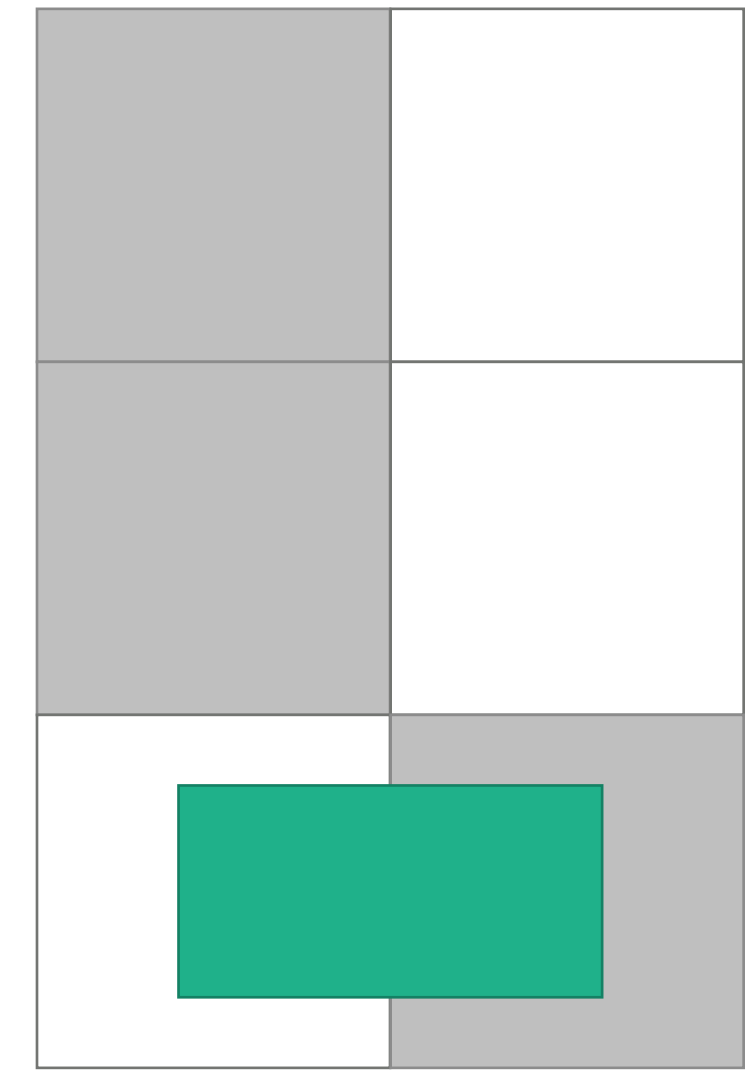
7 0



6 1



5 2



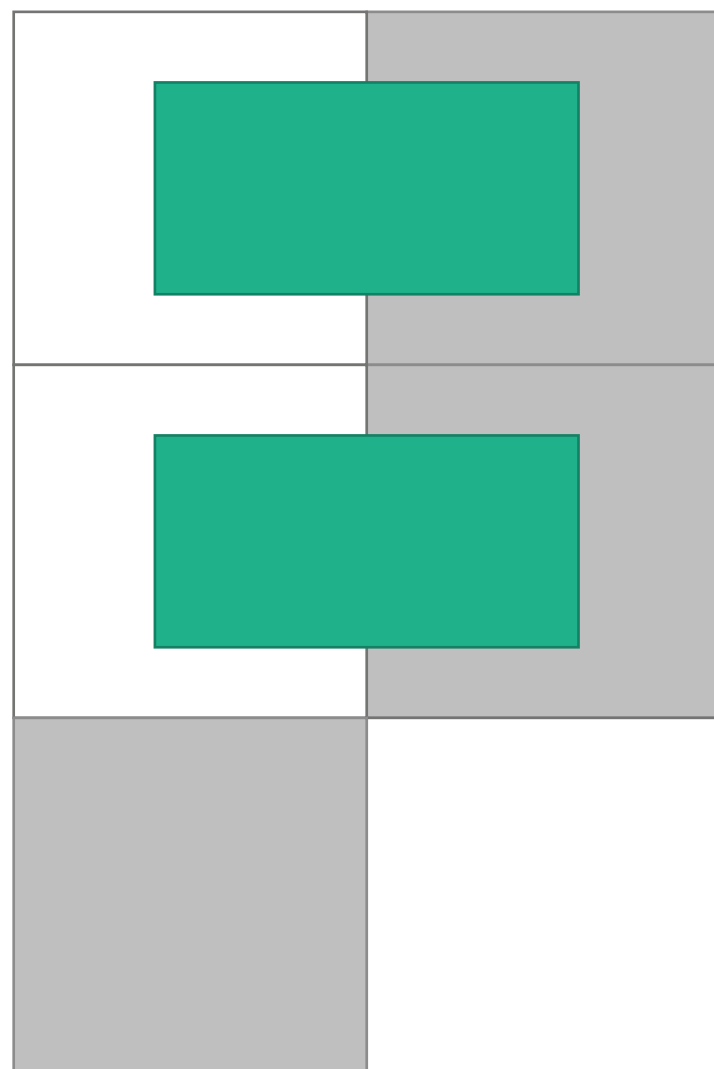
3 4

타일 채우기

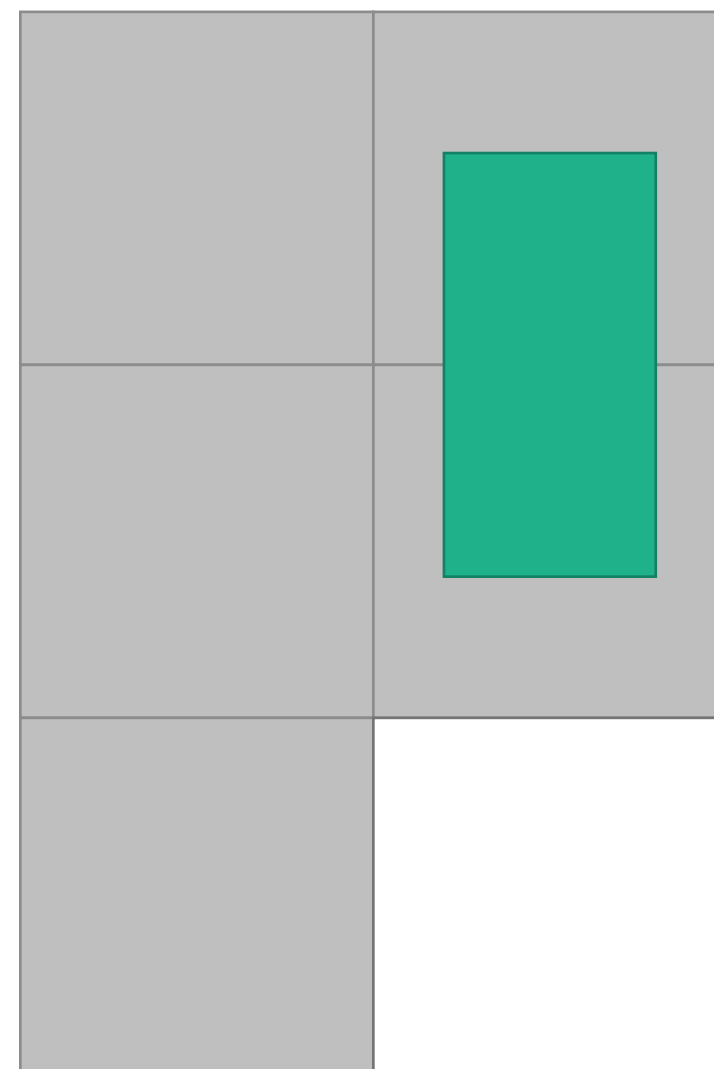
5

<https://www.acmicpc.net/problem/2133>

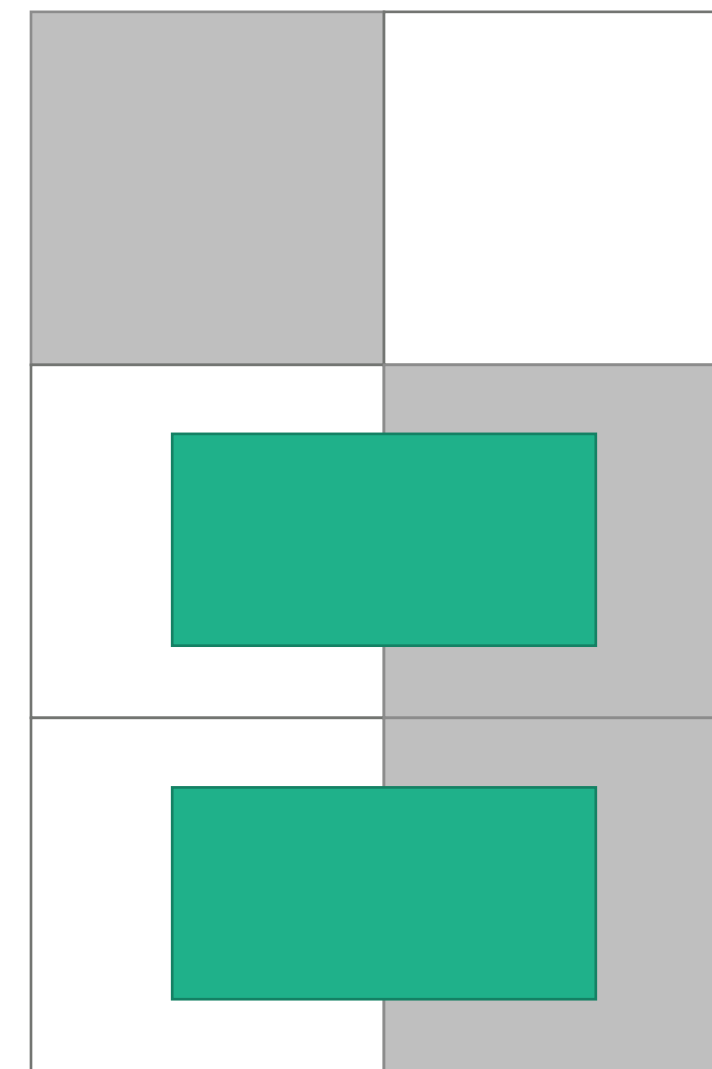
- i 열을 채울 때, $i-1$ 에 빈 칸이 있으면 안된다



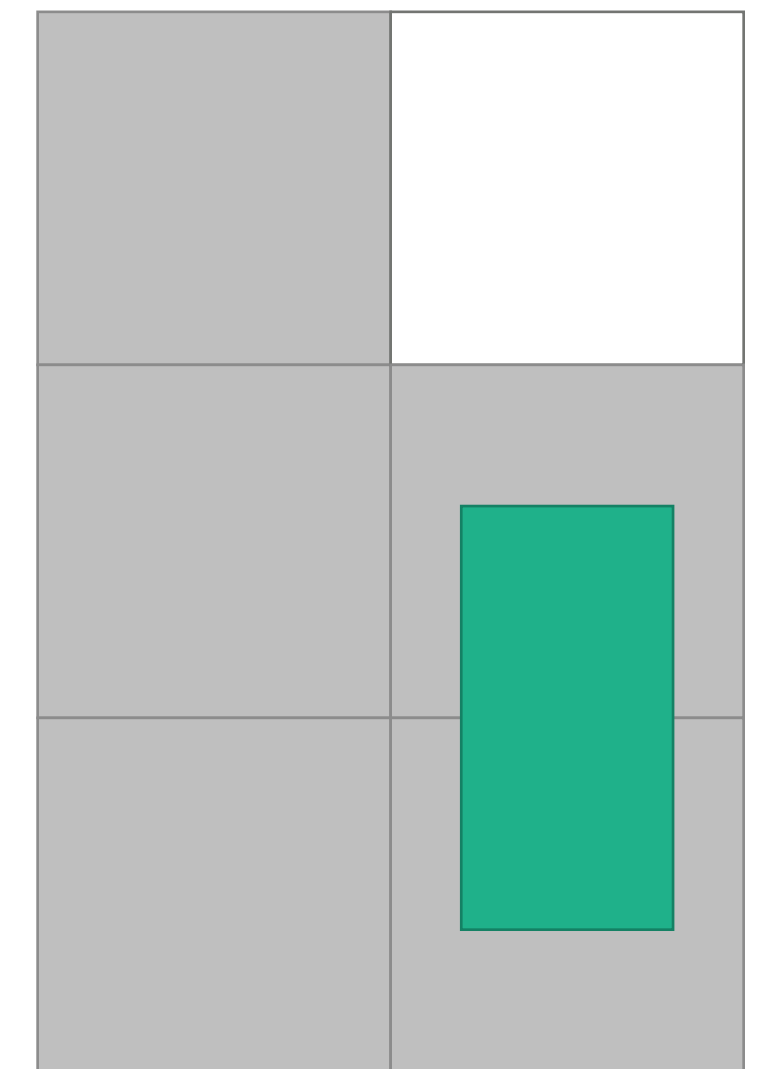
4 3



7 3



1 6

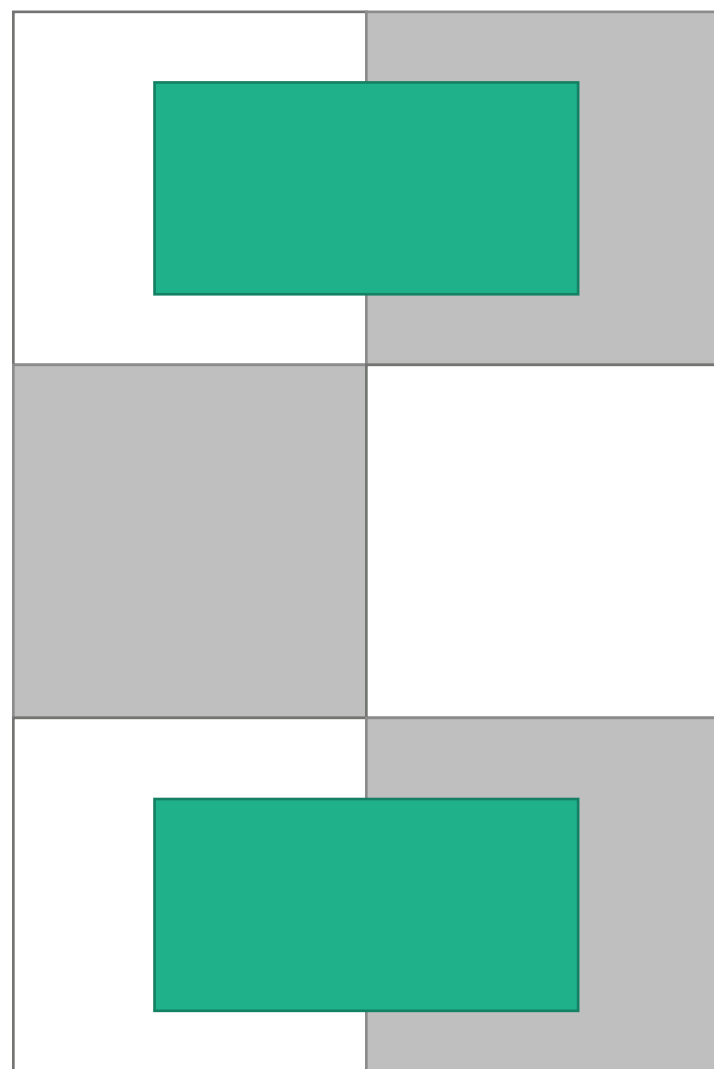


7 6

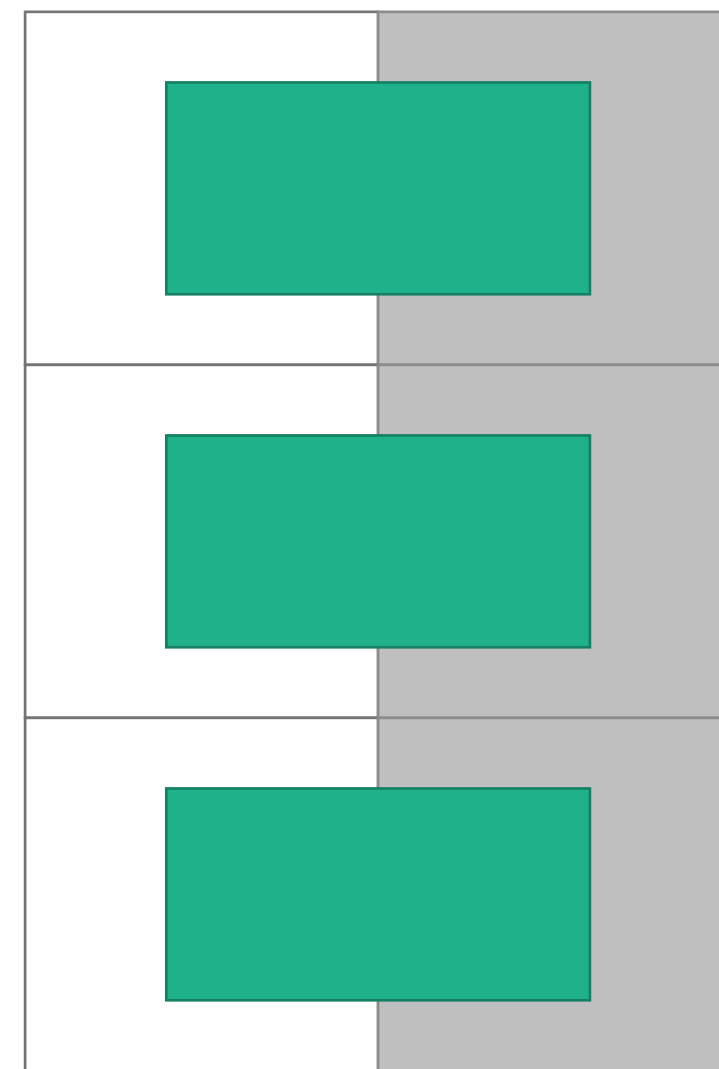
타일 채우기

<https://www.acmicpc.net/problem/2133>

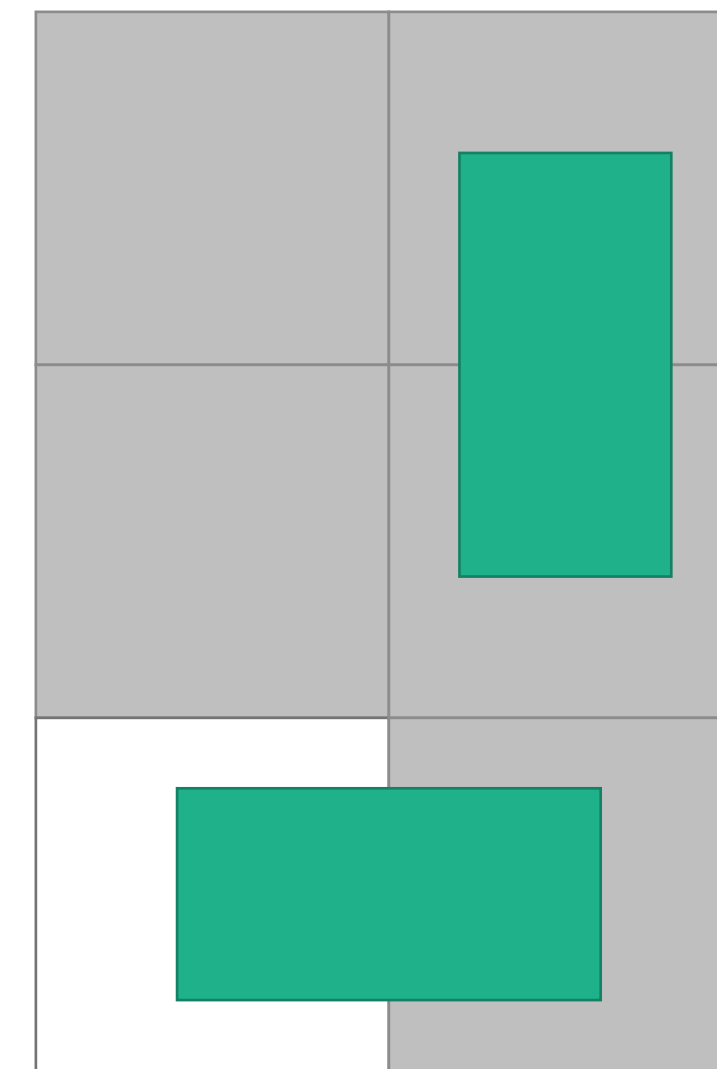
- i 열을 채울 때, $i-1$ 에 빈 칸이 있으면 안된다



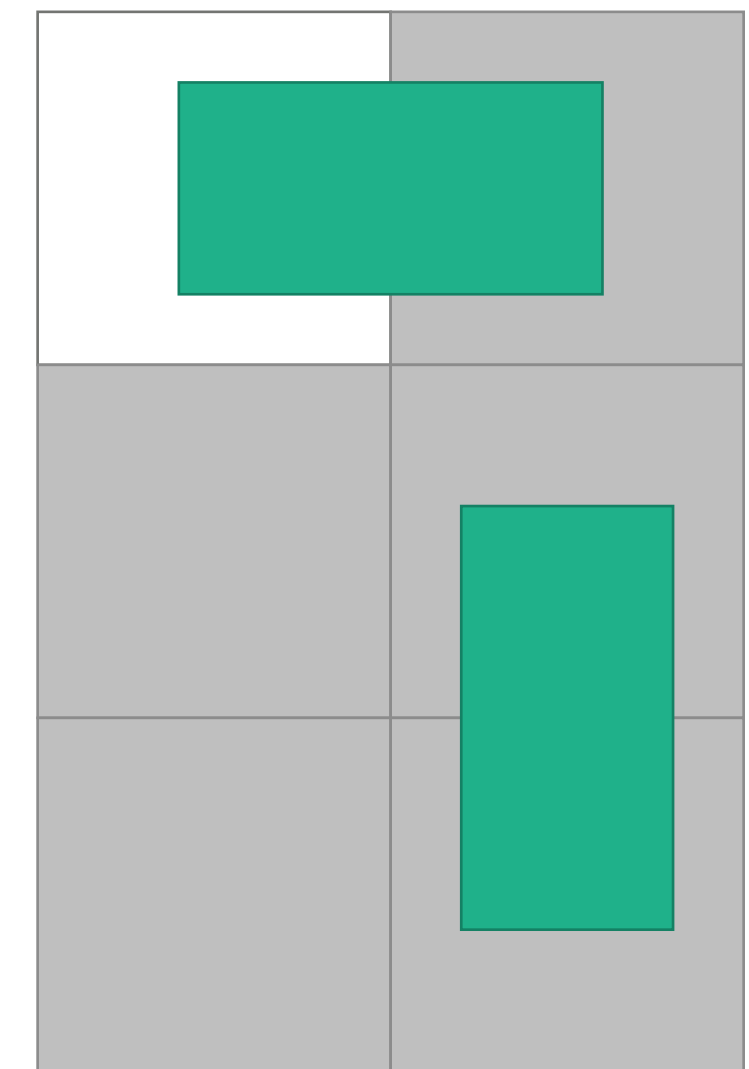
2 5



0 7



3 7



6 7

타일 채우기

<https://www.acmicpc.net/problem/2133>

- $D[i][0] = D[i-1][7]$
- $D[i][1] = D[i-1][6]$
- $D[i][2] = D[i-1][5]$
- $D[i][4] = D[i-1][3]$
- $D[i][3] = D[i-1][4] + D[i-1][7]$
- $D[i][6] = D[i-1][1] + D[i-1][7]$
- $D[i][5] = D[i-1][2]$
- $D[i][7] = D[i-1][0] + D[i-1][3] + D[i-1][6]$

타일 채우기

<https://www.acmicpc.net/problem/2133>

```
D[0][7] = 1;
for (int i=1; i<=n; i++) {
    D[i][0] = D[i-1][7];
    D[i][1] = D[i-1][6];
    D[i][2] = D[i-1][5];
    D[i][4] = D[i-1][3];
    D[i][3] = D[i-1][4] + D[i-1][7];
    D[i][6] = D[i-1][1] + D[i-1][7];
    D[i][5] = D[i-1][2];
    D[i][7] = D[i-1][0] + D[i-1][3] + D[i-1][6];
}
```


타일 채우기

<https://www.acmicpc.net/problem/2133>

- 소스: <http://codeplus.codes/5a614cdff44d4cd9a84612b1ad292515>

외판원 순회

10

<https://www.acmicpc.net/problem/2098>

- 도시 1번부터 N번까지 있을 때
- 어느 한 도시에서 출발해서 N개의 도시를 거쳐 다시 원래 도시로 돌아오는 순회 여행 경로
- 비용의 최소값
- $N \leq 16$

외판원 순회

<https://www.acmicpc.net/problem/2098>

- $D[S][i]$ = 도시를 방문한 상태가 S이고, 현재 있는 위치가 i일때 최소값
- 시작 도시는 1로 고정한다.
- $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$
- $2 \rightarrow 3 \rightarrow 1 \rightarrow 2$
- 모두 같기 때문

외판원 순회

<https://www.acmicpc.net/problem/2098>

- 상태를 사용할 수 있는 이유
- $1 \rightarrow 4$ 까지 정답이
- $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ 라고 하자.
- 그러면, $1 \rightarrow 3 \rightarrow 2 \rightarrow 4$ 가 더 크다면, 방문한 도시의 집합은 같기 때문에 필요없는 값이 된다.
- $1 \rightarrow 5$ 로 갈 때 4에서 가는 경우라면
- $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ 가
- $1 \rightarrow 3 \rightarrow 2 \rightarrow 4 \rightarrow 5$ 보다 무조건 최소값이다

외판원 순회

<https://www.acmicpc.net/problem/2098>

- $D[S][i]$ = 도시를 방문한 상태가 S 이고, 현재 있는 위치가 i 일때 최소값

외판원 순회

<https://www.acmicpc.net/problem/2098>

- $D[S][i] = D[S2][j] + A[j][i]$
- $S2 = S$ 에서 i 를 뺀 값
- i 는 S 에는 포함되어 있어야 하고, $S2$ 에는 포함되어 있지 않아야 한다
- j 는 S 에 포함되어 있어야 한다
- 초기값
 - $D[1][0] = 0$
 - 나머지 = MAX
- 정답
 - $\min(D[(1 \ll N) - 1][i] + A[i][0])$

외판원 순회

<https://www.acmicpc.net/problem/2098>

```
d[1][0] = 0;
for (int i=0; i<(1<<n); i++) {
    for (int j=1; j<n; j++) {
        if (i&(1<<j)) {
            for (int k=0; k<n; k++) {
                if (k!=j && (i&(1<<k)) && a[k][j]) {
                    d[i][j] = min(d[i][j], d[i-(1<<j)][k]+a[k][j]);
                }
            }
        }
    }
}
```

외판원 순회

<https://www.acmicpc.net/problem/2098>

- 소스: <http://codeplus.codes/82b935b9bc5b484999fdd1a833dc7bb1>

외판원 순회

<https://www.acmicpc.net/problem/2098>

- $D[S2][j] = D[S][i] + A[i][j]$
- $S2 = S$ 에서 j 를 더한 값
- i 는 S 에는 포함되어 있어야 하고, $S2$ 에는 포함되어 있지 않아야 한다
- j 는 $S2$ 에 포함되어 있어야 한다
- 초기값
 - $D[1][0] = 0$
 - 나머지 = MAX
- 정답
 - $\min(D[(1 \ll N) - 1][i] + A[i][0])$

외판원 순회

<https://www.acmicpc.net/problem/2098>

```
d[1][0] = 0;
for (int i=0; i<(1<<n); i++) {
    for (int j=0; j<n; j++) {
        if (i&(1<<j)) {
            for (int k=0; k<n; k++) {
                if (k!=j && !(i&(1<<k)) && a[j][k]) {
                    d[i|(1<<k)][k] = min(d[i|(1<<k)][k], d[i][j]+a[j][k]);
                }
            }
        }
    }
}
```

외판원 순회

<https://www.acmicpc.net/problem/2098>

- 소스: <http://codeplus.codes/41dd9bd8e0bf43f68cdb0e30eca96678>

계단 수

20

<https://www.acmicpc.net/problem/1562>

- N이 주어질 때, 길이가 N이면서 0에서 9가 모두 등장하는 계단 수가 총 몇 개 있는지 구하기
- $1 \leq N \leq 100$

쉬운 계단 수

<https://www.acmicpc.net/problem/10844>

- 인접한 자리의 차이가 1이 나는 수를 계단 수라고 한다
- 예: 45656
- 길이가 N인 계단 수의 개수를 구하는 문제

쉬운 계단 수

<https://www.acmicpc.net/problem/10844>

- $D[i][j]$ = 길이가 i 이고 마지막 숫자가 j 인 계단 수의 개수
- $D[i][j] = D[i-1][j-1] + D[i-1][j+1]$

계단 수

<https://www.acmicpc.net/problem/1562>

- $D[N][M][S]$ = 길이가 N이고, M으로 끝나는 계단수, 지금까지 나온 수의 집합: S
- $D[N+1][M+1][S \mid (1 \ll (M+1))] += D[N][M][S]$
- $D[N+1][M-1][S \mid (1 \ll (M-1))] += D[N][M][S]$

계단 수

<https://www.acmicpc.net/problem/1562>

- 소스: <http://codeplus.codes/680c93a9e91743b18142c733cef586d5>

계단 수

25

<https://www.acmicpc.net/problem/1562>

- 이 문제는 계단수 이기 때문에, 상태 S 를 범위로 나타낼 수 있다.
- $D[N][M][L][R]$ = 길이가 N 이고, M 으로 끝나는 계단수, 지금까지 나온 수: $L \sim R$

계단 수

26

<https://www.acmicpc.net/problem/1562>

- 소스: <http://codeplus.codes/0defb3cf202d4593ae999f8a2d3396f5>

컨닝

<https://www.acmicpc.net/problem/1014>

- N행 M열 직사각형 교실에서 시험을 보려고 한다.
- 최대 몇 명의 학생이 시험을 볼 수 있는가?
- $1 \leq N, M \leq 10$

X		X
X	사람	X

컨닝

<https://www.acmicpc.net/problem/1014>

- $D[i][j]$ = i 번 행의 상태가 j 일 때, 앞을 수 있는 학생의 수

X		X
X	사람	X

컨닝

<https://www.acmicpc.net/problem/1014>

- $D[i][j]$ = i 번 행의 상태가 j 일 때, 앉을 수 있는 학생의 수
- $D[i][j] = \max(D[i-1][k] + (j \text{ 상태에 있는 학생의 수}))$
 - k 와 j 에서 서로 못 앉는 자리가 있으면 안됨

X		X
X	사람	X

컨닝

<https://www.acmicpc.net/problem/1014>

- 소스: <http://codeplus.codes/332e1e62f2c2480db4a4e75fc6088596>

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $N \times M$ 격자판을 2×1 크기의 도미노로 채우는 방법의 수
- $1 \leq N, M \leq 14$

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $D[num][s]$ = num번 칸을 채울 것이고, num번 칸부터 M개의 상태가 s일 때, 경우의 수

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $D[num][s]$ = num번 칸을 채울 것이고, num번 칸부터 M개의 상태가 s일 때, 경우의 수
- $i = 7$ 인 경우 상태 s가 나타내는 범위 7~12번 칸 (M개 칸)

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $D[num][s]$ = num번 칸을 채울 것이고, num번 칸부터 M개의 상태가 s일 때, 경우의 수
- num = 7인 경우 상태 s가 나타내는 범위 7~12번 칸 (M개 칸)
- s = 4인 경우: 4는 2진수로 000100(2) 이다. 따라서, 9번 칸만 이미 채워져 있는 상태

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $D[num][s]$ = num번 칸을 채울 것이고, num번 칸부터 M개의 상태가 s일 때, 경우의 수
- num = 7인 경우 상태 s가 나타내는 범위 7~12번 칸 (M개 칸)
- s = 5인 경우: 5는 2진수로 000101(2) 이다. 따라서, 7, 9번 칸이 이미 채워져 있는 상태

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $D[num][s]$ = num번 칸을 채울 것이고, num번 칸부터 M개의 상태가 s일 때, 경우의 수
- num = 7인 경우 상태 s가 나타내는 범위 7~12번 칸 (M개 칸)
- s = 13인 경우: 13은 2진수로 001101(2) 이다. 따라서, 7, 9, 10번 칸이 이미 채워져 있는 상태

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- $D[num][s]$ = num번 칸을 채울 것이고, num번 칸부터 M개의 상태가 s일 때, 경우의 수
- num = 7인 경우 상태 s가 나타내는 범위 7~12번 칸 (M개 칸)
- s = 0인 경우: 0은 2진수로 000000(2) 이다. 따라서, 모두 비어있는 상태

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- 왜 M개를 저장할까?
- num번 칸에 블록을 놓는 경우에 항상 num번 칸이 왼쪽 또는 위가 되게 놓기 때문

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- 상태 s 에서 이미 1로 되어있는 곳은 아래 그림과 같이 위에서 채웠다는 의미

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- 블록을 놓을 수 없는 경우
- 7이 이미 채워져 있는 경우
- 상태 s 를 이진수로 나타냈을 때, 마지막 비트가 1인 경우 $((s \& 1) == 1)$
- 이런 경우에는 채울 수 없기 때문에, 다음 칸을 채워야 함
- s 를 오른쪽으로 한 비트 shift 해야함 $D[num+1][s>>1]$

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- 블록을 놓을 수 있는 경우 (1x2를 놓는 경우)
- $s \& 1$ 과 $s \& 2$ 모두 0이어야 함
- 이 때, 가장 오른쪽 칸인지 아닌지 확인하는 절차도 필요
- $D[num+2][s \gg 2]$

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

42

<https://www.acmicpc.net/problem/1648>

- 블록을 놓을 수 있는 경우 (2x1를 놓는 경우)
- 현재 칸이 비어있으면 항상 가능함
- $D[num+1][(s \gg 1) | (1 \ll (M-1))]$

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17

격자판 채우기

<https://www.acmicpc.net/problem/1648>

- 올바르게 채웠는지는 어떻게 확인할까?
- $N \times M$ 번째 칸이 존재한다고 가정
- $N \times M$ 번째 칸에서 상태가 S이면 올바르게 채운 것!

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18					

격자판 채우기

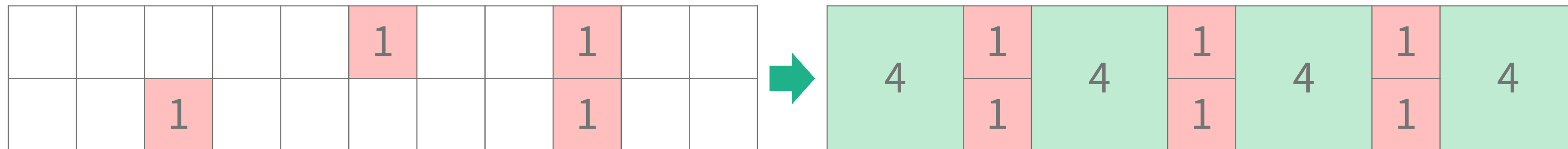
<https://www.acmicpc.net/problem/1648>

- 소스: <http://codeplus.codes/53b943d072564fa19fc27369f73d8db8>

4블럭

<https://www.acmicpc.net/problem/14389>

- 1블럭의 크기는 1×1 , 점수는 1점
- 4블럭의 점수는 2×2 , 점수는 16점
- 1블럭이 일부 채워진 $N \times M$ 크기의 보드가 주어졌을 때, 점수의 최대값을 구하는 문제
- $1 \leq N \leq 10, 1 \leq M \leq 25$



4블럭

<https://www.acmicpc.net/problem/14389>

```
if (a[i][j] == '1') {
    return ans = go(index+1, state >> 1) + 1;
}
ans = go(index+1, state>>1);
if ((state & 1) == 0) {
    ans = max(ans, go(index+1, state>>1) + 1);
}
if (i != n-1 && j != m-1 && (state & 1) == 0 && (state & 2) == 0) {
    if (a[i][j+1] == '.' && a[i+1][j] == '.' && a[i+1][j+1] == '.') {
        ans = max(ans, go(index+2, (state>>2) | (1<<(m-1)) | (1<<(m-2)))) + 16);
    }
}
```

4블럭

<https://www.acmicpc.net/problem/14389>

- 소스: <http://codeplus.codes/765f67db770548799f71e1c893f99726>

체스판

<https://www.acmicpc.net/problem/12960>

- 윤호는 L-모양으로 생긴 타일을 매우 많이 가지고 있다.
- 윤호는 동혁이의 체스판에 다음 조건을 만족시키면서 타일을 체스판 위에 올려 놓으려고 한다.
 - 모든 타일은 회전 시킬 수 있다. (90, 180, 270도)
 - 모든 타일은 체스판 위의 세 칸을 덮어야 한다.
 - 타일은 겹치면 안된다.
 - 말이 이미 올려져있는 칸은 타일이 덮을 수 없다.
 - 타일의 꼭지점 칸(두 정사각형과 붙어있는 칸)은 체스판의 검정 칸을 덮어야 한다.
- 윤호가 놓을 수 있는 타일의 최대 개수를 구하는 문제
- 체스판의 크기: $R \times C$ ($1 \leq R \leq 4, 1 \leq C \leq 47$)

체스판

<https://www.acmicpc.net/problem/12960>

```
// BW
// W.
if (j+1 < m && i+1 < n) {
    if ((state&2) == 0 && a[i][j+1] == '.' && a[i+1][j] == '.') {
        ans = max(ans, go(index+2, (state>>2) | (1<<(m-2))) + 1);
    }
}
// WB
// .W
if (j+1 < m && i+1 < n) {
    if ((state&2) == 0 && a[i][j+1] == '.' && a[i+1][j+1] == '.') {
        ans = max(ans, go(index+2, (state>>2) | (1<<(m-1))) + 1);
    }
}
```

체스판

50

<https://www.acmicpc.net/problem/12960>

```
// .W
// WB
if (j-1 >= 0 && i+1 < n) {
    if ((state&(1<<(m-1))) == 0 && a[i+1][j-1] == '.' && a[i+1][j] == '.') {
        ans = max(ans, go(index+1, (state>>1) | (1<<(m-1)) | (1<<(m-2)))) + 1);
    }
}

// W.
// BW
if (i+1 < n && j+1 < m) {
    if (a[i+1][j] == '.' && a[i+1][j+1] == '.') {
        ans = max(ans, go(index+2, (state>>2) | (1<<(m-1)) | (1<<(m-2)))) + 1);
    }
}
```

체스판

51

<https://www.acmicpc.net/problem/12960>

- 소스: <http://codeplus.codes/bed3885b9945440f812b421629c6f4ad>

타일 놓기

<https://www.acmicpc.net/problem/14390>

- $N \times M$ 크기의 바닥이 있다. ($1 \leq N, M \leq 10$)
- 바닥을 직사각형 모양의 타일로 가득 채우려고 한다.
- 타일의 크기는 $1 \times k$ (k 는 임의의 양의 정수)
- 최소 타일 개수를 구하는 문제

타일 놓기

<https://www.acmicpc.net/problem/14390>

- $D[i][state]$ = i 행의 상태가 $state$ 일 때, 타일의 최소 개수
- $D[i][state] = \min(D[i-1][prev_state]) + (i\text{번 행에 새로 놓는 타일의 개수})$

타일 놓기

<https://www.acmicpc.net/problem/14390>

```
int cnt = 0; int last = -100;
// 0 : -, 1: |
for (int k=0; k<m; k++) {
    if (a[i][k] == '#') continue;
    if (cur & (1<<k)) {
        if (i-1 == 0) cnt++;
        else if ((prev & (1<<k)) == 0) cnt++;
        else if (a[i-1][k] == '#') cnt++;
    } else {
        if (last+1 != k) cnt++;
        last = k;
    }
}
```

타일 놓기

<https://www.acmicpc.net/problem/14390>

- 소스: <http://codeplus.codes/3e51d3d9a81944a7a9d95b787453025a>

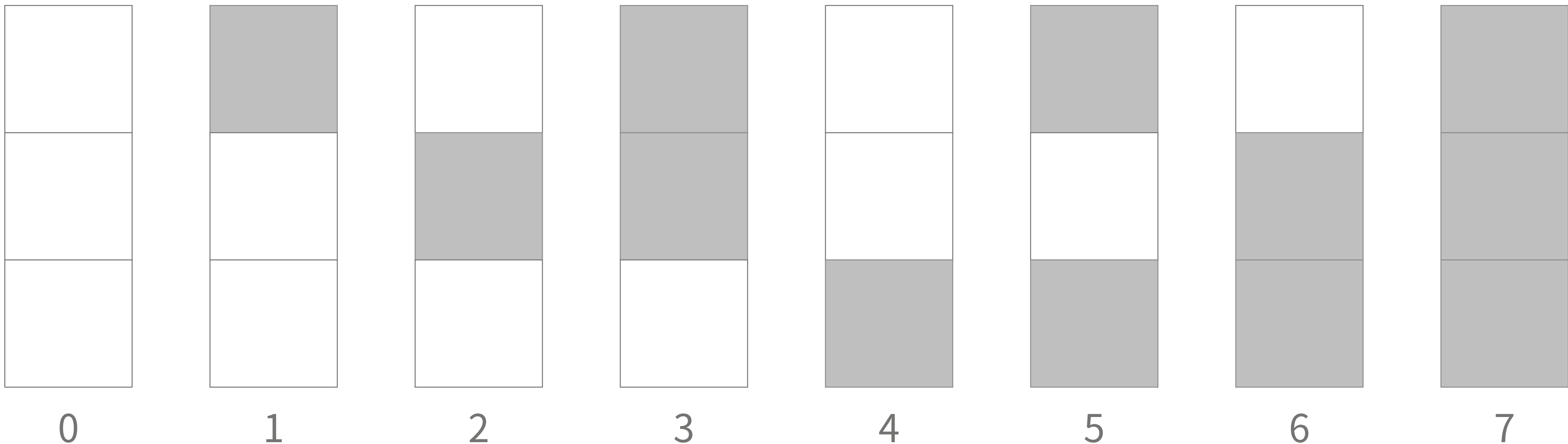
행렬과 다이나믹

타일 채우기 2

57

<https://www.acmicpc.net/problem/13976>

- $3 \times N$ 을 1×2 , 2×1 로 채우는 방법의 수 ($1 \leq N \leq 10^{18}$)
- $D[i][j] = 3 \times i$ 를 채우는 방법의 수, i 열의 상태는 j
- 마지막에 올 수 있는 가능한 경우의 수 (회색: 채워져 있는 칸)



타일 채우기 2

<https://www.acmicpc.net/problem/13976>

- $D[i][0] = D[i-1][7]$
- $D[i][1] = D[i-1][6]$
- $D[i][2] = D[i-1][5]$
- $D[i][4] = D[i-1][3]$
- $D[i][3] = D[i-1][4] + D[i-1][7]$
- $D[i][6] = D[i-1][1] + D[i-1][7]$
- $D[i][5] = D[i-1][2]$
- $D[i][7] = D[i-1][0] + D[i-1][3] + D[i-1][6]$

타일 채우기 2

59

<https://www.acmicpc.net/problem/13976>

$$\begin{bmatrix} D[i][0] \\ D[i][1] \\ D[i][2] \\ D[i][3] \\ D[i][4] \\ D[i][5] \\ D[i][6] \\ D[i][7] \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} D[i-1][0] \\ D[i-1][1] \\ D[i-1][2] \\ D[i-1][3] \\ D[i-1][4] \\ D[i-1][5] \\ D[i-1][6] \\ D[i-1][7] \end{bmatrix}$$

타일 채우기 2

60

<https://www.acmicpc.net/problem/13976>

- 소스: <http://codeplus.codes/593bc03934d545feae8b152bf5ab2354>

정수 수열

<https://www.acmicpc.net/problem/14440>

- A_0, A_1, x, y 가 주어졌을 때, A_N 을 구하는 문제
- $A_N = x \times A_{N-1} + y \times A_{N-2}$
- $1 \leq x, y \leq 99, 0 \leq N < 10^8$

정수 수열

<https://www.acmicpc.net/problem/14440>

- $A_N = x \times A_{N-1} + y \times A_{N-2}$
- $\begin{pmatrix} A_N \\ A_{N-1} \end{pmatrix} = \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} A_{N-1} \\ A_{N-2} \end{pmatrix}$
- $1 \leq x, y \leq 99, 0 \leq N < 10^8$

정수 수열

<https://www.acmicpc.net/problem/14440>

- $A_N = x \times A_{N-1} + y \times A_{N-2}$
- $\begin{pmatrix} A_N \\ A_{N-1} \end{pmatrix} = \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} A_{N-1} \\ A_{N-2} \end{pmatrix}$
- $\begin{pmatrix} A_N \\ A_{N-1} \end{pmatrix} = \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} A_{N-1} \\ A_{N-2} \end{pmatrix}$
- $1 \leq x, y \leq 99, 0 \leq N < 10^8$

정수 수열

<https://www.acmicpc.net/problem/14440>

- $A_N = x \times A_{N-1} + y \times A_{N-2}$
- $\begin{pmatrix} A_N \\ A_{N-1} \end{pmatrix} = \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} A_{N-1} \\ A_{N-2} \end{pmatrix}$
- $\begin{pmatrix} A_N \\ A_{N-1} \end{pmatrix} = \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix} \times \begin{pmatrix} A_{N-1} \\ A_{N-2} \end{pmatrix}$
- $\begin{pmatrix} A_N \\ A_{N-1} \end{pmatrix} = \begin{pmatrix} x & y \\ 1 & 0 \end{pmatrix}^{N-1} \times \begin{pmatrix} A_1 \\ A_0 \end{pmatrix}$
- $1 \leq x, y \leq 99, 0 \leq N < 10^8$

정수 수열

65

<https://www.acmicpc.net/problem/14440>

- 소스: <http://codeplus.codes/144e71630e534a9eb56991aff493a231>

정수 수열

<https://www.acmicpc.net/problem/14440>

- $A_N = x \times A_{N-1} + y \times A_{N-2}$
- 항상 이전 두 개의 값에 의존하기 때문에, 두 개를 이용해서 주기를 찾을 수 있다.

정수 수열

<https://www.acmicpc.net/problem/14440>

- $x = 9, y = 4, A_0 = 51, A_1 = 33$
- 51, 33, 1, 41, 73, 21, 81, 13, 41, 21, 53, 61, 61, 93, 81, 1, 33, 1, 41, 73, 21, 81

정수 수열

68

<https://www.acmicpc.net/problem/14440>

- $x = 9, y = 4, A_0 = 51, A_1 = 33$

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
51	33	1	41	73	21	81	13	41	21	53	61	61	93	81	1	33	1	41	73	21	81	13	41	21

정수 수열

69

<https://www.acmicpc.net/problem/14440>

- $x = 9, y = 4, A_0 = 51, A_1 = 33$

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
51	33	1	41	73	21	81	13	41	21	53	61	61	93	81	1	33	1	41	73	21	81	13	41	21

주기 = 15

주기 = 15

정수 수열

70

<https://www.acmicpc.net/problem/14440>

- 소스: <http://codeplus.codes/1d8f46acc09d4a28a3843da773a563c7>