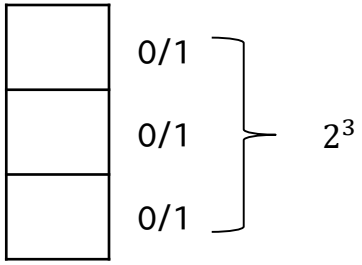
타일 채우기

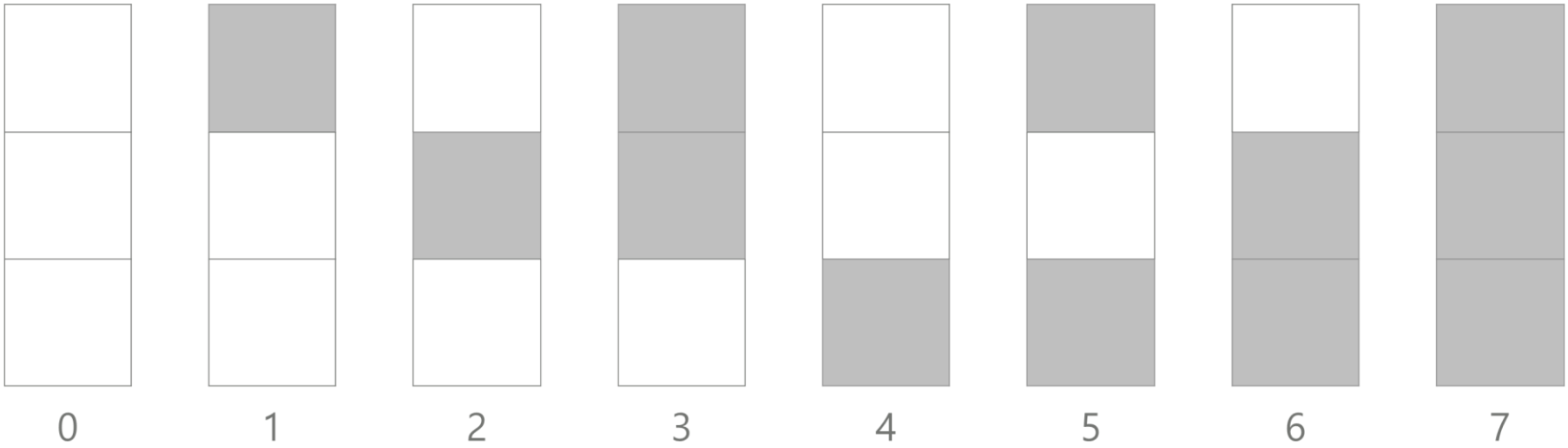
https://www.acmicpc.net/problem/2133

1 : 채워짐,
0 : 안 채워짐



- 3 x N을 1 x 2, 2 x 1로 채우는 방법의 수이다.
- dp[i][j] = 3 x i를 채우는 방법의 수, i열의 상태는 j
- 마지막에 올 수 있는 가능한 경우의 수 (회색 : 채워져 있는 칸)

- 8가지 상태는 아래와 같다.



Gaussian

타일 채우기

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

i열을 채울 때, i-1에 빈 칸이 있으면 안된다.

① 0번 상태 앞에는 7번 상태만 올 수 있다.

② 1번 상태 앞에는 6번 상태만 올 수 있다.

③ 2번 상태 앞에는 5번 상태만 올 수 있다.

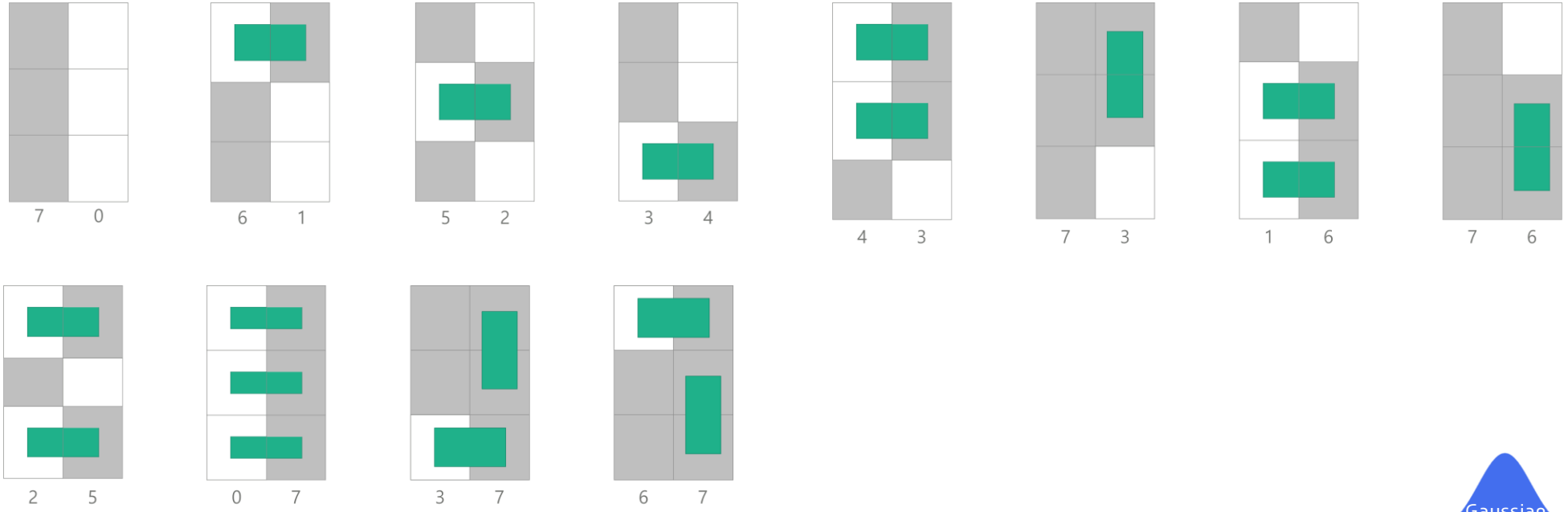
④ 3번 상태 앞에는 4, 7번 상태만 올 수 있다.

⑤ 4번 상태 앞에는 3번 상태만 올 수 있다.

⑥ 5번 상태 앞에는 2번 상태만 올 수 있다.

⑦ 6번 상태 앞에는 1, 7번 상태만 올 수 있다.

⑧ 7번 상태 앞에는 0, 3, 6번 상태만 올 수 있다.



Gaussian

타일 채우기

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

- 3 x N을 1 x 2, 2 x 1로 채우는 방법의 수이다.
- dp[i][j] = 3 x i를 채우는 방법의 수, i열의 상태는 j
- dp[i][0] = dp[i-1][7];
- dp[i][1] = dp[i-1][6];
- dp[i][2] = dp[i-1][5];
- dp[i][3] = dp[i-1][4] + dp[i-1][7];
- dp[i][4] = dp[i-1][3];
- dp[i][5] = dp[i-1][2];
- dp[i][6] = dp[i-1][1] + dp[i-1][7];
- dp[i][7] = dp[i-1][0] + dp[i-1][3] + dp[i-1][6];

Gaussian

타일 채우기

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

```
int solve(int i, int j) {
    if (i == 0 && j == 7) return 1;
    if (i == 0) return 0;

    if (dp[i][j] >= 0) return dp[i][j];

    dp[i][j] = 0;
    if (j == 0) dp[i][j] += solve(i - 1, 7);
    if (j == 1) dp[i][j] += solve(i - 1, 6);
    if (j == 2) dp[i][j] += solve(i - 1, 5);
    if (j == 3) dp[i][j] += solve(i - 1, 4) + solve(i - 1, 7);
    if (j == 4) dp[i][j] += solve(i - 1, 3);
    if (j == 5) dp[i][j] += solve(i - 1, 2);
    if (j == 6) dp[i][j] += solve(i - 1, 1) + solve(i - 1, 7);
    if (j == 7) dp[i][j] += solve(i - 1, 0) + solve(i - 1, 3) + solve(i - 1, 6);

    return dp[i][j];
}
```

Gaussian