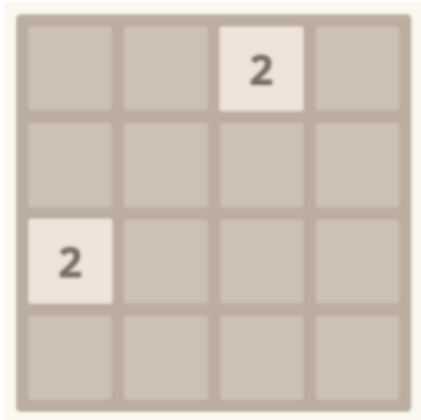
완전탐색 + 구현 

알고리즘 스터디 5주차 강수지

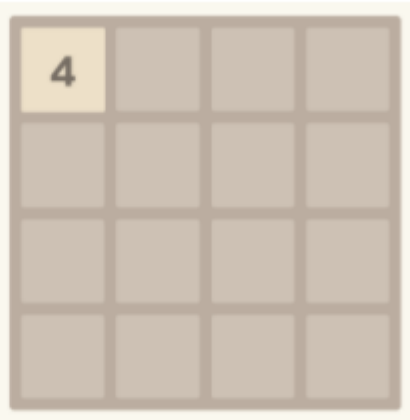
백준 12100 : 2048 (Easy)

https://www.acmicpc.net/problem/12100



<그림 1>

<그림 2>



<그림 3>

- N*N 크기 보드 위에서 전체 블록을 상하좌우 네 방향 중 하나로 이동
- 두 블록이 충돌하면 하나로 합쳐진다
- 한번의 이동에서 이미 합쳐진 블록은 또 다른 블록과 합쳐질 수 없다
- 최대 5번 이동으로 가장 큰 블록의 값을 출력하는 문제

백준 12100 : 2048 (Easy) 접근방법

1) 주어진 조건에 맞춰 **구현**

- 한번에 한 방향으로만 이동하며 모든 블록이 같이 움직인다
- 이동 과정에 같은 값을 가진 블록을 만나면
  이동 방향에 있는 블록 위치로 값이 합쳐진다
- 이동 과정에서 한번 합쳐진 블록은 다시 같은 값을 만나도 합쳐질 수 없다

2) 입력 조건으로 주어진 “ 보드의 크기 N (1 ≤ N ≤ 20)“ 조건을 통해 **완전탐색** !

백준 12100 : 2048 (Easy) 풀이

```
from collections import deque
n = int(input())
board = [list(map(int, input().split())) for _ in range(n)]
answer, q = 0, deque()

def solve(count):
    global board, answer
    if count == 5:
        for i in range(n):
            answer = max(answer, max(board[i]))
        return
    b = [x[:] for x in board]

    for k in range(4):
        move(k)
        solve(count+1)
        board = [x[:] for x in b]

def get(i, j):
    if board[i][j]:
        q.append(board[i][j])
        board[i][j] = 0
```

- 입력
  - 보드 n*n board 입력

- solve 함수 – 이동횟수
  - 최대 5번 이동 후 최대값 반환
  - 4 방향으로 움직임
  - 원본 board 저장

- get 함수 – board 값
  - 0이 아닌 경우 큐에 저장
  - 저장 후 0 으로 처리

백준 12100 : 2048 (Easy) 풀이

```
def merge(i, j, di, dj):
    while q:
        x = q.popleft()
        if not board[i][j]:
            board[i][j] = x
        elif board[i][j] == x:
            board[i][j] = x*2
            i, j = i+di, j+dj
        else:
            i, j = i+di, j+dj
            board[i][j] = x
```

- merge 값 더하기

- 행, 열, y 방향, x 방향
- 움직이려는 블록 값
- 0과 값이 다른 경우 그대로
- 값이 일치하는 경우 * 2배

백준 12100 : 2048 (Easy) 풀이

```
def move(k):
    if k == 0:
        for j in range(n):
            for i in range(n):
                get(i, j)
            merge(0, j, 1, 0)
    elif k == 1:
        for j in range(n):
            for i in range(n-1, -1, -1):
                get(i, j)
            merge(n-1, j, -1, 0)
    elif k == 2:
        for i in range(n):
            for j in range(n):
                get(i, j)
            merge(i, 0, 0, 1)
    else:
        for i in range(n):
            for j in range(n-1, -1, -1):
                get(i, j)
            merge(i, n-1, 0, -1)
```

- move 이동

- 위로 이동하는 경우

맨 위까지 이동한 경우에 row index 는 0

위로 올라오기때문에 row index가 1씩 증가

- 아래로 이동하는 경우

맨 아래까지 이동한 경우에 row index 는 n-1

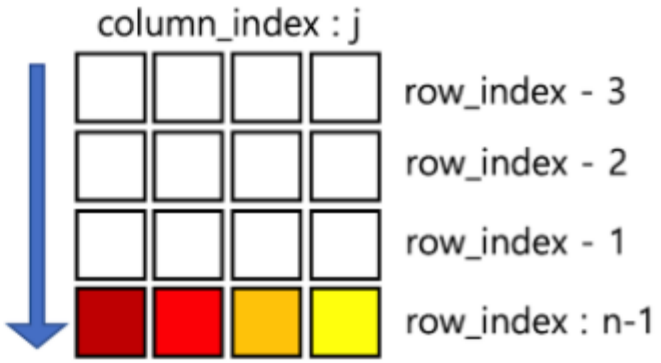
아래로 내려가기때문에 row index가 -1씩

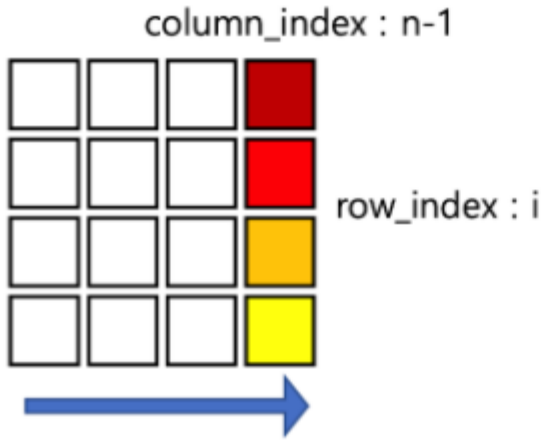
- 오른쪽으로 이동하는 경우

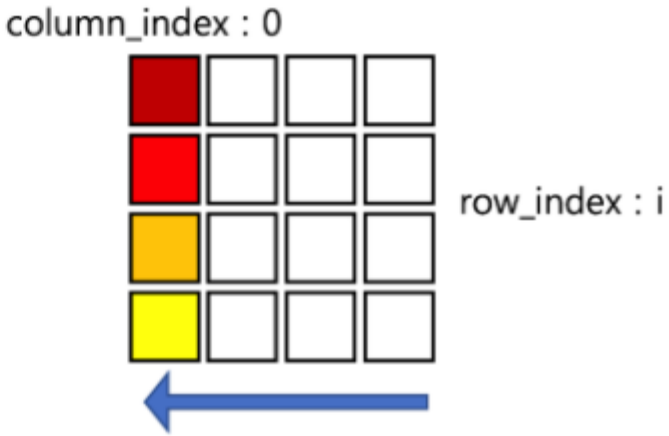
맨 오른쪽으로 이동한 경우 column index 0

Column index + 1 (왼쪽의 경우 -1)

백준 12100 : 2048 (Easy) 풀이







move 함수 이해하기 !

- 위 아래로 움직일 때 column index 고정 row index 만 움직임

- 오른쪽 왼쪽으로 움직일 때 row index 고정 column index 만 고정

백준 14501 : 퇴사

https://www.acmicpc.net/problem/14501

오늘부터 N+1일째 되는 날 퇴사를 하기 위해서, 남은 N일 동안 최대한 많은 상담을 하려고 한다.

백준이는 비서에게 최대한 많은 상담을 잡으라고 부탁을 했고, 비서는 하루에 하나씩 서로 다른 사람의 상담을 잡아놓았다.

각각의 상담은 상담을 완료하는데 걸리는 기간 T_i와 상담을 했을 때 받을 수 있는 금액 P_i로 이루어져 있다.

N = 7인 경우에 다음과 같은 상담 일정표를 보자.

	1일	2일	3일	4일	5일	6일	7일
T_i	3	5	1	1	2	4	2
P_i	10	20	10	20	15	40	200

- N일 동안 상담의 최대 이익을 구하는 문제

백준 14501 : 퇴사 접근방법

1) 완전 탐색으로 푸는 경우

- 각 상담 일자에 상담이 가능한 경우를 모두 테스트
- 결과 중 가장 수익이 높은 경우를 출력
- 입력 조건으로 주어진 “$N (1 \le N \le 15)$“ 조건을 통해 **완전탐색** !

2) DP 풀이 가능

- n번째 일자에 상담이 끝났다면 n-1 일까지 중 상담을 최대로 진행하여 n번째 일자에 상담이 끝나는 경우의 이익을 구하는 것
- 1 ~ n+1 일자에 상담이 끝나는 경우 모두 동일하게 적용
- 작은 문제가 동일하게 반복되며 그 결과가 큰 문제 해결 **DP**

백준 14501 풀이

- 입력

- earn 함수 – 상담날짜 이익
- 해당 날짜의 상담을 기간안에 진행이 가능할 경우
- 해당 날짜의 상담을 기간안에 진행할 수 없는 경우

```python
import sys
input = sys.stdin.readline

N = int(input())

schedule = [list(map(int, input().split())) for _ in range(N)]

answer = 0
def earn(day, money):
    global answer
    answer = max(answer, money)

    if day >= N:
        return

    if day + schedule[day][0] <= N:
        earn(day + schedule[day][0], money + schedule[day][1])
        earn(day + 1, money)
    else:
        earn(day + 1, money)
    return

earn(0, 0)
print(answer)
```

공통문제

- 백준 16637 괄호추가하기 (골드 3)
- 백준 17136 색종이 붙이기 (골드 2)
- 백준 17406 배열 돌리기 4 (골드 4)

참고 사이트

https://swexpertacademy.com/main/main.do

감사합니다