SAMSUNG SW AI ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

# 삼성청년 SW·AI아카데미

## SWEA5653 줄기세포배양

SAMSUNG SW AI ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

# 줄기세포배양

# 목표

K 시간 후 살아있는 줄기세포의 총 개수를 구하는 프로그램 작성하기

# 문제 요약

1. 줄기 세포는 **비활성 상태**로 시작 → **일정 시간(생명력)이 지난 뒤 활성화**
2. 활성화된 세포는 **상,하,좌,우** 4방향으로 번식한다.
3. 새로운 세포는 **부모 세포의 생명력**을 가진다.
4. 활성화 된 세포는 **생명력만큼 시간이 지나면 죽는다**.
5. 같은 시간에 같은 위치로 번식하는 세포가 있을 경우, **생명력이 더 높은 세포가 우선**이다.
6. K 시간이 지난 후, **비활성+활성 상태의 세포 수를 출력**해야 한다.

# 제약 사항



## 초기 줄기 세포가 분포된 영역의 넓이 ( N X M )

- 1≤N≤50, 1≤M≤50

## 배양 시간 ( K )

- 1≤K≤300

## 배양 용기의 크기는 무한하다.

- 줄기 세포가 배양 용기 가장자리에 닿아서 번식할 수 없는 경우는 없다.

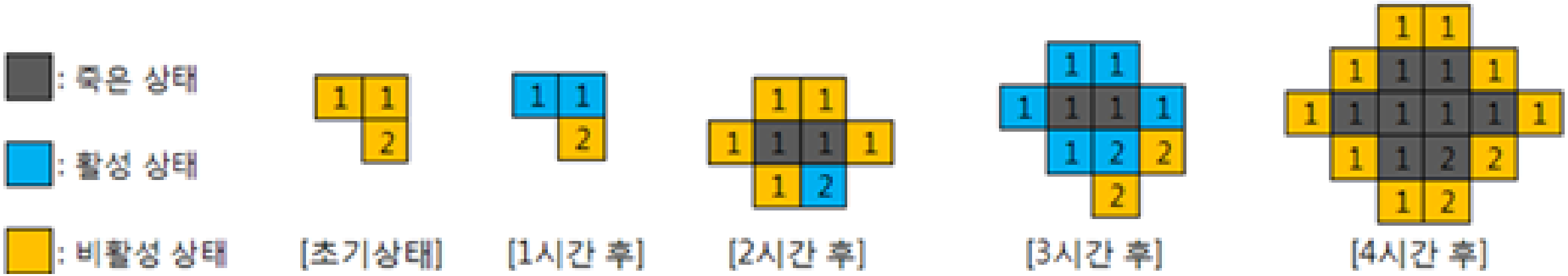
## 줄기 세포의 생명력 ( X )

- 1≤X≤10

# 전반적인 흐름이 bfs를 이용한 flood fill과 유사하다.

- **반복문을 이용해서 배양시간(K) 초 만큼 반복하며 시뮬레이션을 한다.**
- **다만! 생명력이 다른 세포끼리 번식하는 영역이 겹치면??**

  → 생명력이 큰 쪽이 차지하므로, 우선 순위가 존재한다. ( priority queue 사용 )



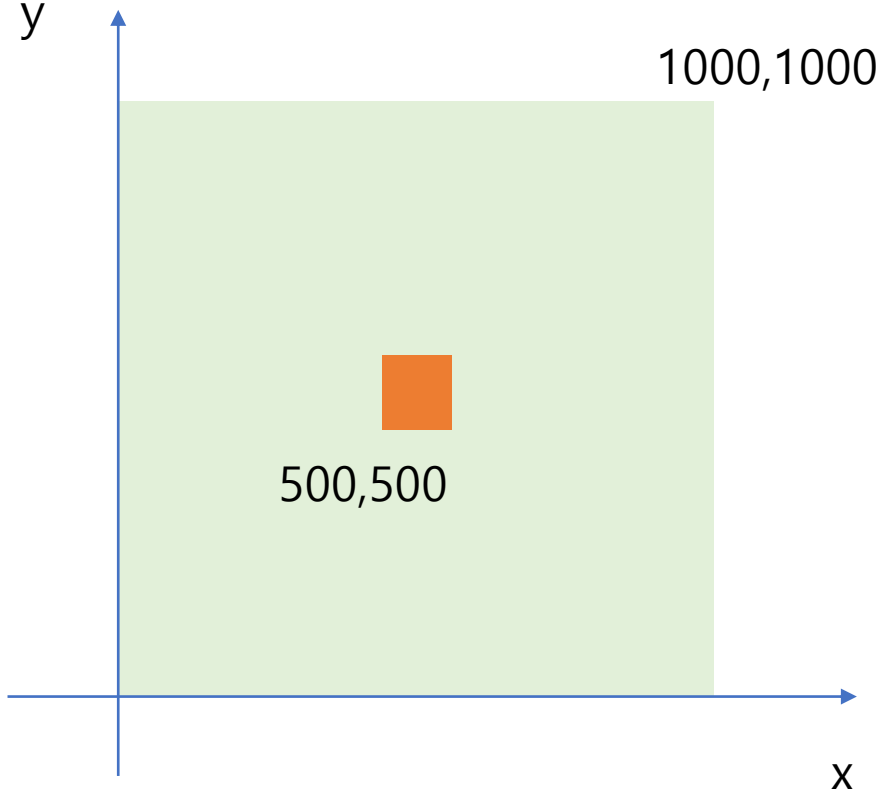
# 1. 배양 용기 크기는 무한하다. → 과연 그럴까?

- 초기 상태의 줄기 세포 분포 영역 크기가 50x50 이고, ( 1 ≤ N ≤ 50 , 1 ≤ M ≤ 50 )
- 배양 시간(K)은 최대 300시간이기 때문에, 아무리 번식을 잘해도 넉넉히 줘야 1000x1000 이 최대이다.
- 줄기 세포 생명력 최대 10, 세포의 번식 영역이 겹칠 경우, 우선 순위가 있기 때문에,

최대 케이스가 나오지 않아 넉넉히 1000x1000으로도 진행 가능

- 초기 배양 위치도 500, 500 만큼 쉬프트해서 설정한다.

  → 0,0 부터 받으면, 음수 범위로까지 확장이 된다.



# 데이터 입력 받기



T : 테스트 케이스

N M K : 세로 크기, 가로 크기, 배양 시간

N x M 그리드 상태 정보

```
//초기 영역 N, M
//배양시간 K
int N, M, K;
//배양용기 크기
int map[1000][1000];
```

```
for (int tc = 1; tc <= T; tc++) {

    cin >> N >> M >> K;
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < M; j++) {
            //시작 좌표를 map 중앙으로 이동 +500 시프트
            int y = i + 500;
            int x = j + 500;
            cin >> map[y][x];
        }
    }

    int ans = 0;
    cout << "#" << tc << " " << ans << '₩n';
}
```

# 2. 세포의 정보 관리 ( 구조체 사용 )

- 세포의 정보는 생명력( X ) 만 직접적으로 들어온다.
- 세포는 X 시간 뒤에 활성화 되고, X 만큼 살아 있다가, X 시간 뒤에 죽는다.

- 전체 프로그램을 0~K(배양시간) 초까지 시뮬레이션을 하면서,
- 세포들이 각자의 생명력에 맞춰 대기 → 활성화 → 죽음 순으로 진행한다.

- 해당 시간에 맞춰 세포가 동작할 수 있게,

생명력, 활동 시간, 죽는 시간 을 같이 관리한다.

세포의 개수는 특정되지 않으므로, vector로 관리한다.

세포

```
int y
int x
int life
int active_time
int dead_time
```

# 세포 구조체에 정보 입력



입력 데이터를 이용해 세포 구조체에 정보를 채운다.

```
//세포 정보용 구조체
struct NODE {
    int y;
    int x;
    int life;
    int active_time;
    int dead_time;
};
//vector로 세포 관리
vector<NODE> v;
```

```
int y = i + 500;
int x = j + 500;

int life;
cin >> life;
map[y][x] = life;

//생명력이 0이면 vector에 등록할 필요 없다.
if (life == 0) continue;
v.push_back({ y,x,life,life,life + life });
```

# 모든 세포들이 이제 번식을 한다.

- 활성화 시간이 되면? → pq에 넣기
- 죽을 시간이 되면? **카운팅**
  - → 총 세포 수 - 죽은 세포 수 = 살아있는 세포 수
- 번식 ( bfs )

```
solve();
```

```
void solve() {
    //시간 변수
    int t = 1;
    //시뮬레이션 시작

        //모든 세포들 중에서 탐색

            //활성화 시간이 되었다면? 우선순위에 넣기


            //죽을 시간이 되었다면? 카운팅


        //번식 시작

        //시간 흐름

    }
}
```

# 4. bfs 를 이용한 번식 구현



bfs를 이용해 번식한다.

- queue 대신 priority queue 사용
- 초기화도 해야 한다.

```
//bfs로 번식 시작
void bfs(int now_time) {

    while (!pq.empty()) {
        //뽑기


        //각 방향으로 번식 시작

            //다음 좌표


            //비어있지 않으면 넘김

            //번식

            //비활성 세포 추가
            //생명력을 등록하는 시간에 주의한다.


    }
}
```

# [참고] 메모리 계산하기



## 메모리 제한 조건

- **힙, 정적 메모리 합쳐서 262144 kbytes ( 256MB ) 이내, 스택 메모리 1024 kbytes (1MB)이내**

## 내가 짜는 코드의 메모리 계산을 어느 정도 할 순 있어야 한다.

- **배열이나 벡터 하나가 100만 개 이상 되면 “메모리 괜찮나?” 하고 한번 점검하자.**

| 요소 | 하나당 크기 | 1,000,000개일 때 |
|---|---|---|
| int | 4B | 4MB |
| bool | 1B | 1MB |
| pair<int,int> | 8B | 8MB |
| struct (예: Node, 5개 int) | 20B | 20MB |

SAMSUNG SW AI ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

# 내일 방송에서 만나요!

삼성 청년 SW 아카데미