**[제한 사항]**

|  |  |
| --- | --- |
| 시간 | 25개 테스트케이스를 합쳐서 C++ 의 경우 3초 / Java 의 경우 3초 |
| 메모리 | 힙, 정적 메모리 합쳐서 256MB 이내, 스택 메모리 1MB 이내 |

**※** 본 문제는 임직원의 S/W 문제해결 역량 향상을 위한 **Professional 실전형 연습문제**이며

함께 제공되는 **Testcase 또한 학습을 위한 연습용**입니다.

|  |
| --- |
| **①   C 또는 C++로 답안을 작성하시는 응시자께서는 검정시스템에 제출 시, Language 에서 C++ 를 선택하신 후 제출하시기 바랍니다.**  **②   Main 과 User Code 부분으로 구성되어 있습니다.**  **ㅇ   A.**Main         : 수정할 수 없는 코드이며, 채점 시 비 정상적인 답안 검출 등 평가를 위한 로직이 추가 될 수 있습니다.  **ㅇ   B.**User Code  : 실제 응시자가 작성해야 하는 코드이며, 제출 시에는 표준 입출력 함수가 포함되어 있으면 안 됩니다.  **③   Local PC 에서 프로그래밍 시 유의 사항**  **ㅇ   A.**2개의 파일을 생성하셔야 합니다. ( main.cpp / solution.cpp 또는 Solution.java / UserSolution.java )  **ㅇ   B.**Main 부분의 코드를 main.cpp 또는 Solution.java 에 복사해서 사용하시기 바랍니다.  **ㅇ   C.**sample\_input.txt 를 사용하시기 위해서는 Main 부분의 코드 내에  **ㅇ   C.**표준 입력을 파일로 전환하는 코드 ( 주석처리 되어 있음 ) 의 주석을 풀어서 사용하시면 됩니다.  **ㅇ   D.**User Code 부분의 코드를 작성하신 후 서버에 제출하실 때,  **ㅇ   D.**디버깅을 위한 표준 입출력 함수를 모두 삭제 또는 주석 처리해 주셔야 합니다.  **④   문제 내에 제약조건을 모두 명시하지 않으므로 주어지는 코드를 분석하셔야 합니다.**  **⑤   코드는 개발 언어에 따라 상이할 수 있으므로, 작성할 언어를 기준으로 분석하셔야 합니다.** |

**[문제 설명]**

정사각형 모양의 접시에 특수 약품을 떨어뜨려서 세균을 생성, 번식, 그리고 소멸시키는 모습을 관찰하려고 한다.

세균은 생성 약품에 의해 접시 내에 생성되어 활성화라는 과정을 통해 번식을 진행한 후 비활성화 된다.  
그리고, 소멸 약품을 이용하면 생성된 세균들을 소멸시킬 수 있다.

세균의 번식이 어떤 식으로 이루어지는지 자세히 살펴보자.

세균을 번식시킬 수 있는 정사각형 모양의 접시가 주어진다.  
이 접시는 N개의 행과 N개의 열로 나누어 지며 행과 열이 만나는 지점을 셀이라고 한다.

이 셀의 위치는 (row, col)로 나타내며, “에너지 소비량”이라는 정보를 가지고 있다.  
“에너지 소비량’은 세균이 해당 셀에서 생성될 때 필요로 하는 번식 에너지의 소비량이다.

[Fig. 1] (a)는 N이 6인 접시를 보여준다. 각 셀의 “에너지 소비량”는 [Fig. 1] (b)와 같이 나타낼 수 있다.  
위치가 (1, 6)인 셀의 에너지 소비량은 1이며, 위치가 (5, 2)인 셀의 에너지 소비량은 5가 된다.  
텍스트, 시계, 게이지이(가) 표시된 사진

자동 생성된 설명  
**[Fig. 1]**

**생성 약품을 접시의 특정 위치(mRow, mCol)에 떨어뜨리면 세균의 번식이 진행된다.**  
약품을 떨어뜨릴 때 다음 정보가 주어진다.  
- 타겟 세균(mTarget) : 약품에 의해 활성화되는 세균의 종류로 2종류가 존재한다. (파란 세균, 빨간 세균)  
- 번식 에너지(mEnergy) : 약품이 지니고 있는 특수 에너지로 세균 생성에 사용된다.

번식은 다음 3단계로 나누어 진행된다.

**[1단계] 생성 약품 떨어뜨리기**  
1) 약품이 떨어진 셀의 상태에 따라 아래와 같이 진행한다.  
   1-1) 셀이 비어 있으면 타겟 세균과 동일한 종류의 세균을 생성하면서 바로 활성화시킨다.  
        이때 해당 셀의 에너지 소비량만큼 번식 에너지를 소비한다.  
        번식 에너지보다 소비량이 더 크면 번식 에너지의 남은 양은 0이 된다.  
   1-2) 약품이 떨어진 셀에 타겟 세균과 같은 종류의 세균이 존재하면 해당 세균은 활성화 된다.  
   1-3) 약품이 떨어진 셀에 타겟 세균과 다른 종류의 세균이 존재하면  
        다음 단계로 넘어가지 않고 종료된다.  
2) 활성화된 세균들은 **상하좌우로 연결된 같은 종류의 세균들**을 모두 활성화 시킨다.  
   더 이상 활성화 시킬 세균들이 없을 때까지 이 과정은 계속된다.

[Fig. 2]에서 (a)는 (3, 4)에 약품을 떨어뜨렸을 때 진행 과정이고  
(d)는 (4, 5)에 약품을 떨어뜨렸을 때 진행 과정을 보여준다.  
(연한 색의 사각형은 활성화된 세균을 의미한다.)  
텍스트이(가) 표시된 사진

자동 생성된 설명  
**[Fig. 2]**

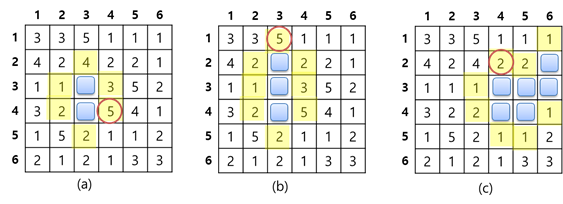
**[2단계] 번식 에너지를 이용한 번식**  
1) 번식 에너지가 0인 경우 3단계로 넘어간다.  
2) 번식 에너지가 존재하는 경우, 활성화된 세균들의 **번식 가능한 위치**들을 모두 찾는다.  
3) 번식 가능한 위치가 하나도 없는 경우 3단계로 넘어간다.  
4) 번식 가능한 위치들 중 **우선순위가 가장 높은 위치**에  
   타겟 세균과 동일한 종류의 세균이 생성되면서 바로 활성화 된다.  
   이때 해당 셀의 에너지 소비량만큼 번식 에너지를 소비한다.  
   번식 에너지보다 소비량이 더 크면 번식 에너지의 남은 양은 0이 된다.  
5) 세균 생성 후 활성화된 세균들은 **상하좌우로 연결된 같은 종류의 세균들**을 모두 활성화 시킨다.  
   더 이상 활성화 시킬 세균들이 없을 때까지 이 과정은 계속된다.  
6) 1)번부터 되풀이 한다.

- **번식 가능한 위치** : 활성화된 세균들의 위치에서 **상하좌우**로 한 칸 이동했을 때 **비어 있는 셀들**을 의미한다.  
  [Fig. 3]에서 노란색 셀들을 의미한다.  
  [Fig. 3] (f)의 경우 비어있는 셀이 주위에 없기 때문에 번식할 수 있는 위치가 없다.

낱말맞추기게임, 점수판이(가) 표시된 사진

자동 생성된 설명  
                                                    **[Fig. 3]**

- **우선 순위** : 비어있는 셀들이 여러 개일 때 아래와 같은 방법으로 우선순위가 가장 높은 셀을 찾는다.  
    1) 셀의 에너지 소비량이 높을수록 우선순위가 더 높다.  
    2) 1)에 해당 하는 셀이 여러 개 있는 경우, 행의 번호가 작은 셀이 우선순위가 더 높다.  
    3) 2)에 해당하는 셀이 여러 개 있는 경우, 열의 번호가 작은 셀이 우선순위가 더 높다.

    [Fig. 4]는 번식 가능한 위치들 중 우선 순위가 가장 높은 셀을 빨간 원으로 표시하고 있다.  
  
                                                     **[Fig. 4]**

**[3단계] 번식의 종료 :** 활성화된 세균들이 모두 비활성화 되면서 번식이 종료된다.

**소멸 약품을 접시의 특정 위치(mRow, mCol)에 떨어뜨리면 세균의 소멸이 진행된다.**

1) 약품이 떨어진 셀의 상태에 따라 아래와 같이 진행한다.  
   1-1) 약품이 떨어진 **셀이 비어있는 경우 세균의 소멸은 진행되지 않고 완료**된다.  
   1-2) 약품이 떨어진 셀에 세균이 존재하면 해당 세균은 활성화 된다.  
2) 활성화된 세균들은 **상하좌우로 연결된 같은 종류의 세균들**을 모두 활성화 시킨다.  
   더 이상 활성화 시킬 세균들이 없을 때까지 이 과정은 계속된다.  
3) 활성화된 세균들은 모두 소멸한다.

[Fig. 5]는 위치 (2, 3)에 소멸 약품을 추가했을 때 세균들이 소멸되는 과정을 보여준다  
약품이 떨어진 위치에 파란 세균이 있었으므로, 주변의 파란 세균들이 모두 소멸되었다.  
텍스트이(가) 표시된 사진

자동 생성된 설명  
                                                     **[Fig. 5]**

위와 같은 특성이 있을 때, 접시에 존재하는 세균의 수를 계산하는 프로그램을 작성하라.

아래 API 설명을 참조하여 각 함수를 구현하라.

※ 아래 함수 signature는 C/C++에 대한 것으로 다른 언어에 대해서는 제공되는 Main과 User Code를 참고하라.

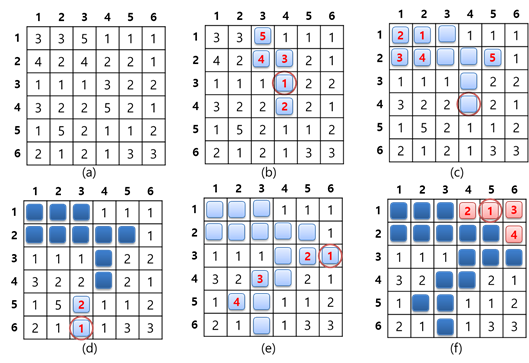
아래는 User Code 부분에 작성해야 하는 API 의 설명이다.

|  |
| --- |
| **void init(int N, int mDish[][])** |
| 각 테스트 케이스의 처음에 호출된다.  접시의 크기가 N행 N열로 주어진다. 초기에는 접시가 비어있다.  접시 내의 위치 (row, col)로 주어지는 셀의 “에너지 소비량”은 **mDish[row-1][col-1]**이다.  예를 들어 위치 (1, 1)의 “에너지 소비량”은 mDish[0][0]이다.    ***Parameters***     N : 접시의 크기 (5 ≤ N ≤ 100)     mDish : 각 셀 내의 에너지 소비량 (1 ≤ mDish[][] ≤ 5) |
| **int dropMedicine(int mTarget, int mRow, int mCol, int mEnergy)** |
| 생성 약품을 위치(mRow, mCol)에 떨어뜨린다.  약품에 대해서 타겟 세균은 mTarget으로 주어지며, 파란 세균은 1로 빨간 세균은 2로 표현된다. 번식 에너지는 mEnergy로 주어진다.  떨어뜨린 약품에 의한 번식이 모두 끝난 후, 접시 내의 살아있는 mTarget과 동일한 종류의 세균의 총 개수를 반환한다.    ***Parameters***     mTarget : 생성할 세균의 종류 (1 ≤ mTarget ≤ 2)     mRow : 약품을 추가할 행 (1 ≤ mRow ≤ N)    mCol : 약품을 추가할 열 (1 ≤ mCol ≤ N)    mEnergy : 번식 에너지의 양 (3 ≤ mEnergy ≤ 200)    ***Return***     접시 내의 종류가 mTarget인 세균의 총 개수 |
| **int cleanBacteria(int mRow, int mCol)** |
| 소멸 약품을 위치(mRow, mCol)에 떨어뜨린다.  소멸이 완료된 후 소멸된 세균과 같은 종류의 살아있는 세균들의 총 개수를 반환한다. 약품이 세균이 없는 위치에 떨어진 경우 -1을 반환한다.  ***Parameters***     mRow : 약품을 추가할 행 (1 ≤ mRow ≤ N)    mCol : 약품을 추가할 열 (1 ≤ mCol ≤ N)    ***Return***     소멸된 세균과 같은 종류의 살아있는 세균들의 총 개수    (소멸된 세균이 없는 경우 -1) |

**[예제]**

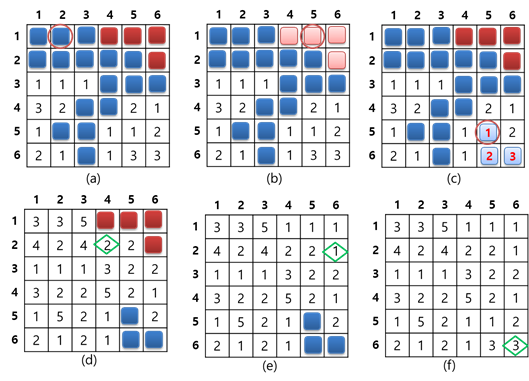
아래에는 테스트케이스 1번을 처음부터 순서대로 보여준다.  
함수 호출 후 세균들이 번식된 모습을 그림으로 확인할 수 있다.  
그림 내의 요소들의 의미는 다음과 같다.  
- 빨간 원 : 생성 약품을 떨어뜨리는 위치  
- 녹색 마름모 : 소멸 약품을 떨어뜨리는 위치  
- 파란 사각형 : mTarget이 1인 세균(파란 세균)  
- 빨간 사각형 : mTarget이 2인 세균(빨간 세균)  
- 세균 안의 빨간 수 : 세균의 생성 순서  
- 생성 약품에 의해 활성화되는 세균들은 연한 색으로 표현한다.

|  |  |  |  |
| --- | --- | --- | --- |
| **순서** | **Function** | **Return** | **Figure** |
| 1 | dish = [            [3, 3, 5, 1, 1, 1],            [4, 2, 4, 2, 2, 1],            [1, 1, 1, 3, 2, 2],            [3, 2, 2, 5, 2, 1],            [1, 5, 2, 1, 1, 2],            [2, 1, 2, 1, 3, 3] ] init(6, dish) |  | [Fig. 6] (a) |
| 2 | dropMedicine(1, 3, 4, 19) | **5** | [Fig. 6] (b) |
| 3 | dropMedicine(1, 4, 4, 14) | **10** | [Fig. 6] (c) |
| 4 | dropMedicine(1, 6, 3, 3) | **12** | [Fig. 6] (d) |
| 5 | dropMedicine(1, 3, 6, 10) | **16** | [Fig. 6] (e) |
| 6 | dropMedicine(2, 1, 5, 20) | **4** | [Fig. 6] (f) |

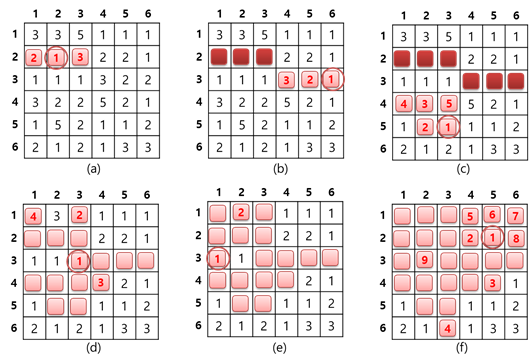
  
                                                   **[Fig. 6]**

|  |  |  |  |
| --- | --- | --- | --- |
| 7 | dropMedicine(2, 1, 2, 7) | **4** | [Fig. 7] (a) |
| 8 | dropMedicine(2, 1, 5, 5) | **4** | [Fig. 7] (b) |
| 9 | dropMedicine(1, 5, 5, 5) | **19** | [Fig. 7] (c) |
| 10 | cleanBacteria(2, 4) | **3** | [Fig. 7] (d) |
| 11 | cleanBacteria(2, 6) | **0** | [Fig. 7] (e) |
| 12 | cleanBacteria(6, 6) | **0** | [Fig. 7] (f) |

순서 8에서 번식할 빈 셀이 존재하지 않는다.  
순서 10에서 파란색 세균이 소멸 되었으므로, 살아있는 파란색 세균의 개수를 반환한다.

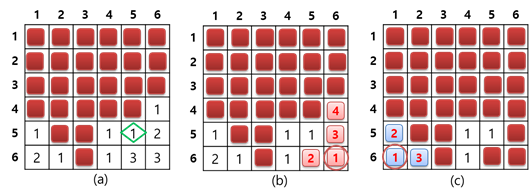
  
                                                     **[Fig.7]**

|  |  |  |  |
| --- | --- | --- | --- |
| 13 | dropMedicine(2, 2, 2, 7) | **3** | [Fig. 8] (a) |
| 14 | dropMedicine(2, 3, 6, 7) | **6** | [Fig. 8] (b) |
| 15 | dropMedicine(2, 5, 3, 13) | **11** | [Fig. 8] (c) |
| 16 | dropMedicine(2, 3, 3, 13) | **15** | [Fig. 8] (d) |
| 17 | dropMedicine(2, 3, 1, 3) | **17** | [Fig. 8] (e) |
| 18 | dropMedicine(2, 2, 5, 13) | **26** | [Fig. 8] (f) |

  
                                                       **[Fig.8]**

|  |  |  |  |
| --- | --- | --- | --- |
| 19 | cleanBacteria(5, 5) | **-1** | [Fig. 9] (a) |
| 20 | dropMedicine(2, 6, 6, 9) | **30** | [Fig. 9] (b) |
| 21 | dropMedicine(1, 6, 1, 20) | **3** | [Fig. 9] (c) |

순서 19에서 소멸 약품을 떨어뜨린 위치에 세균이 존재하지 않음으로 -1을 반환한다.

  
                                                        **[Fig.9]**

**[제약사항]**

1. 각 테스트 케이스 시작 시 init() 함수가 호출된다.  
2. 각 테스트 케이스에서 dropMedicine() 함수의 호출 횟수는 1,500 이하이다.  
3. 각 테스트 케이스에서 cleanBacteria() 함수의 호출 횟수는 500 이하이다.  
4. 각 테스트 케이스에서 약품을 떨어뜨렸을 때마다 활성화되는 세균들의 개수는 1,000개이하 임을 보장한다.

**[입출력]**

입출력은 제공되는 Main 부분의 코드에서 처리하므로 User Code 부분의 코드에서는 별도로 입출력을 처리하지 않는다.

Sample input 에 대한 정답 출력 결과는 “#TC번호 결과” 의 포맷으로 보여지며 결과가 100 일 경우 정답, 0 일 경우 오답을 의미한다.