

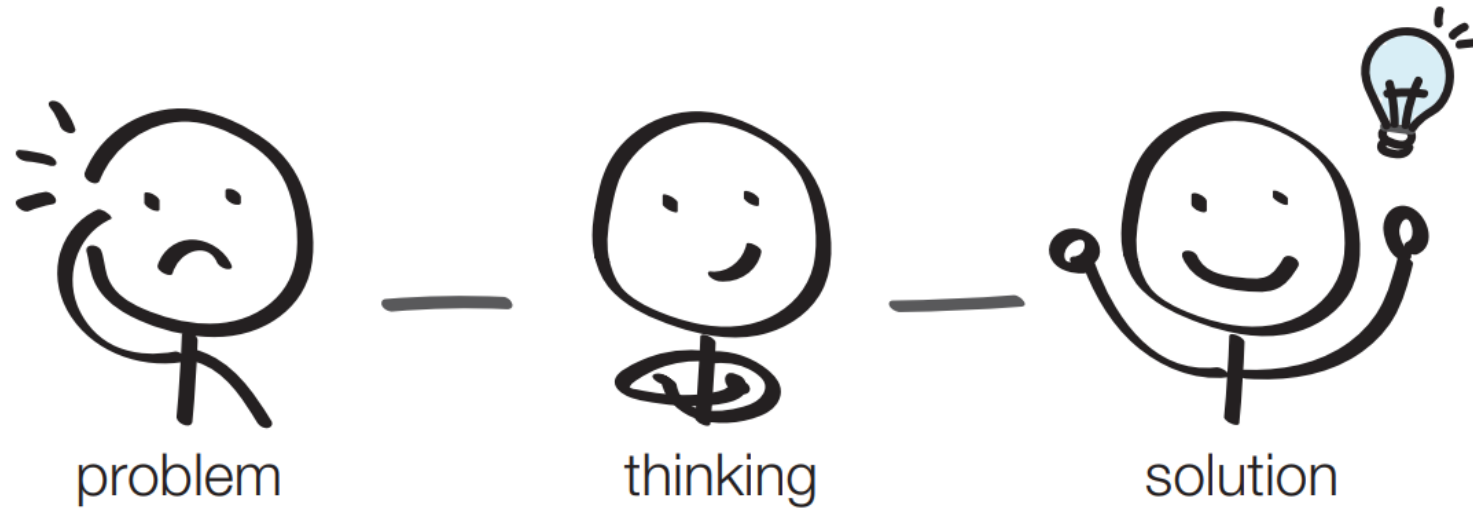


Algorithm

동양미래대학교 강은영

구현

- 구현이란, 머릿속에 있는 알고리즘을 소스코드로 바꾸는 과정입니다.

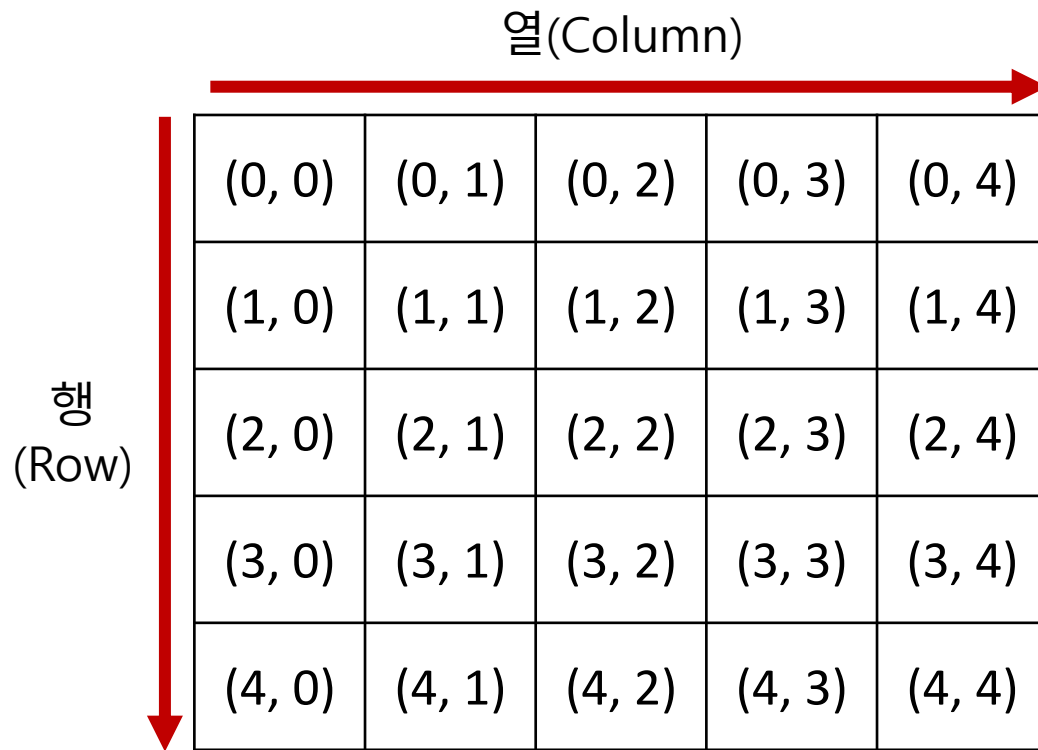


구현(Implementation)

- 흔히 알고리즘 대회에서 구현 유형의 문제란 무엇을 의미할까요?
 - 문제 풀이를 떠올리는 것은 쉽지만 소스코드로 옮기기 어려운 문제를 지칭합니다.
- 구현 유형의 예시는 다음과 같습니다.
 - 알고리즘은 간단한데 코드가 지나칠 만큼 길어지는 문제
 - 실수 연산을 다루고, 특정 소수점 자리까지 출력해야 하는 문제
 - 문자열을 특정한 기준에 따라서 끊어 처리해야 하는 문제
 - 적절한 라이브러리를 찾아서 사용해야 하는 문제

구현(Implementation)

- 일반적으로 알고리즘 문제에서의 2차원 공간은 행렬(Matrix)의 의미로 사용됩니다.



The diagram shows a 5x5 matrix with rows and columns indexed from 0 to 4. A red arrow labeled '열(Column)' points to the right above the matrix, and another red arrow labeled '행 (Row)' points downwards to the left of the matrix.

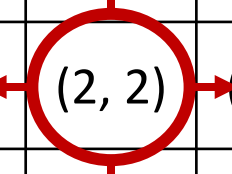
(0, 0)	(0, 1)	(0, 2)	(0, 3)	(0, 4)
(1, 0)	(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 0)	(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 0)	(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 0)	(4, 1)	(4, 2)	(4, 3)	(4, 4)

```
for i in range(5):  
    for j in range(5):  
        print('(', i, ',', j, ')', end=' ')  
    print()
```

구현(Implementation)

- 시뮬레이션 및 완전 탐색 문제에서는 2차원 공간에서의 방향 벡터가 자주 활용됩니다.

(0, 0)	(0, 1)	(0, 2)	(0, 3)	(0, 4)
(1, 0)	(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 0)	(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 0)	(3, 1)	(3, 2)	(3, 3)	(3, 4)
(4, 0)	(4, 1)	(4, 2)	(4, 3)	(4, 4)



```
# 동, 북, 서, 남  
dx = [0, -1, 0, 1]  
dy = [1, 0, -1, 0]
```

```
# 현재 위치  
x, y = 2, 2
```

```
for i in range(4):  
    # 다음 위치  
    nx = x + dx[i]  
    ny = y + dy[i]  
    print(nx, ny)
```

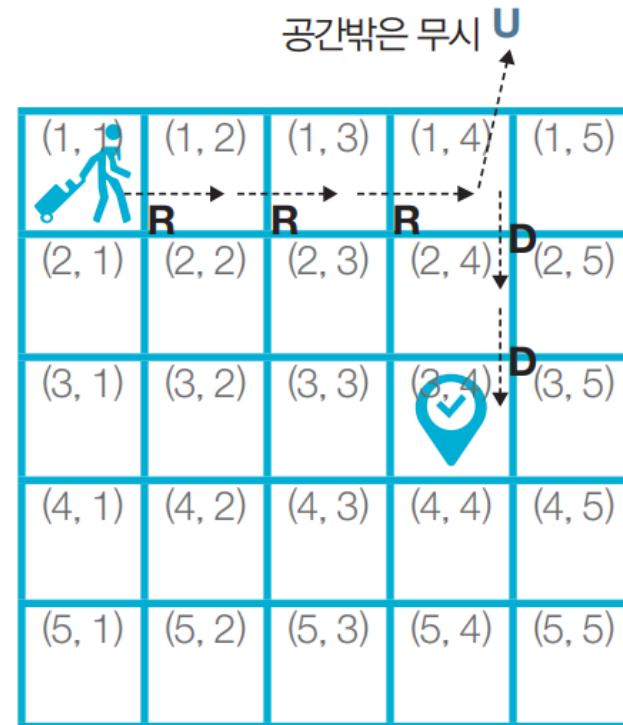
<문제>상하좌우 : 문제 설명

- 여행가 A는 $N \times N$ 크기의 정사각형 공간 위에 서 있습니다. 이 공간은 1×1 크기의 정사각형으로 나누어져 있습니다. 가장 왼쪽 위 좌표는 (1, 1)이며, 가장 오른쪽 아래 좌표는 (N, N)에 해당합니다. 여행가 A는 상, 하, 좌, 우 방향으로 이동할 수 있으며, 시작 좌표는 항상 (1, 1)입니다. 우리 앞에는 여행가 A가 이동할 계획이 적힌 계획서가 놓여 있습니다.
- 계획서에는 하나의 줄에 띄어쓰기를 기준으로 하여 L, R, U, D 중 하나의 문자가 반복적으로 적혀 있습니다. 각 문자의 의미는 다음과 같습니다.
 - L: 왼쪽으로 한 칸 이동
 - R: 오른쪽으로 한 칸 이동
 - U: 위로 한 칸 이동
 - D: 아래로 한 칸 이동

<문제>상하좌우 : 문제 설명

- 이때 여행가 A가 $N \times N$ 크기의 정사각형 공간을 벗어나는 움직임은 무시됩니다. 예를 들어 (1, 1)의 위치에서 L 혹은 U를 만나면 무시됩니다. 다음은 $N = 5$ 인 지도와 계획서입니다.

계획서와 지도
 $R \rightarrow R \rightarrow R \rightarrow U \rightarrow D \rightarrow D$



<문제>상하좌우 : 문제 조건

난이도 ●○○ | 풀이 시간 15분 | 시간제한 2초 | 메모리 제한 128MB

입력 조건

- 첫째 줄에 공간의 크기를 나타내는 N 이 주어집니다. ($1 \leq N \leq 100$)
- 둘째 줄에 여행가 A 가 이동할 계획서 내용이 주어집니다. ($1 \leq$ 이동 횟수 ≤ 100)

출력 조건

- 첫째 줄에 여행가 A 가 최종적으로 도착할 지점의 좌표 (X, Y)를 공백을 기준으로 구분하여 출력합니다.

입력 예시

```
5  
R R R U D D
```

출력 예시

```
3 4
```

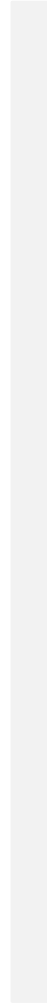

<문제>상하좌우 : 문제 해결 아이디어

- 이 문제는 요구사항대로 충실히 구현하면 되는 문제입니다.
- 일련의 명령에 따라서 개체를 차례대로 이동시킨다는 점에서 시뮬레이션(Simulation) 유형으로도 분류되며 구현이 중요한 대표적인 문제 유형입니다.
 - 다만, 알고리즘 교재나 문제 풀이 사이트에 따라서 다르게 일컬을 수 있으므로, 코딩 테스트에서의 시뮬레이션 유형, 구현 유형, 완전 탐색 유형은 서로 유사한 점이 많다는 정도로만 기억합시다.

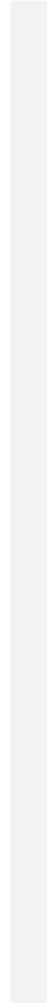
<문제>상하좌우 : 답안 예시(Python)



<문제>상하좌우 : 답안 예시(C++)



<문제>상하좌우 : 답안 예시(Java)



<문제>시각 : 문제 설명

- 정수 N 이 입력되면 00시 00분 00초부터 N 시 59분 59초까지의 모든 시각 중에서 3이 하나라도 포함되는 모든 경우의 수를 구하는 프로그램을 작성하세요. 예를 들어 1을 입력했을 때 다음은 3이 하나라도 포함되어 있으므로 세어야 하는 시각입니다.
 - 00시 00분 03초
 - 00시 13분 30초
- 반면에 다음은 3이 하나도 포함되어 있지 않으므로 세면 안 되는 시각입니다.
 - 00시 02분 55초
 - 01시 27분 45초

<문제>시각 : 문제 조건

난이도 ●○○ | 풀이 시간 15분 | 시간제한 2초 | 메모리 제한 128MB

입력 조건

- 첫째 줄에 정수 N 이 입력됩니다. ($0 \leq N \leq 23$)

출력 조건

- 00시 00분 00초부터 N 시 59분 59초까지의 모든 시각 중에서 3이 하나라도 포함되는 모든 경우의 수를 출력합니다.

입력 예시

5

출력 예시

11475

<문제>시각 : 문제 해결 아이디어

- 이 문제는 가능한 모든 시각의 경우를 하나씩 모두 세서 풀 수 있는 문제입니다.
- 하루는 86,400초이므로, 00시 00분 00초부터 23시 59분 59초까지의 모든 경우는 86,400가지입니다.
 - $24 * 60 * 60 = 86,400$
- 따라서 단순히 시각을 1씩 증가시키면서 3이 하나라도 포함되어 있는지를 확인하면 됩니다.
- 이러한 유형은 완전 탐색(Brute Forcing) 문제 유형이라고 불립니다.
 - 가능한 경우의 수를 모두 검사해보는 탐색 방법을 의미합니다.

<문제>시각 : 답안 예시(Python)



<문제>시각 : 답안 예시(C++)

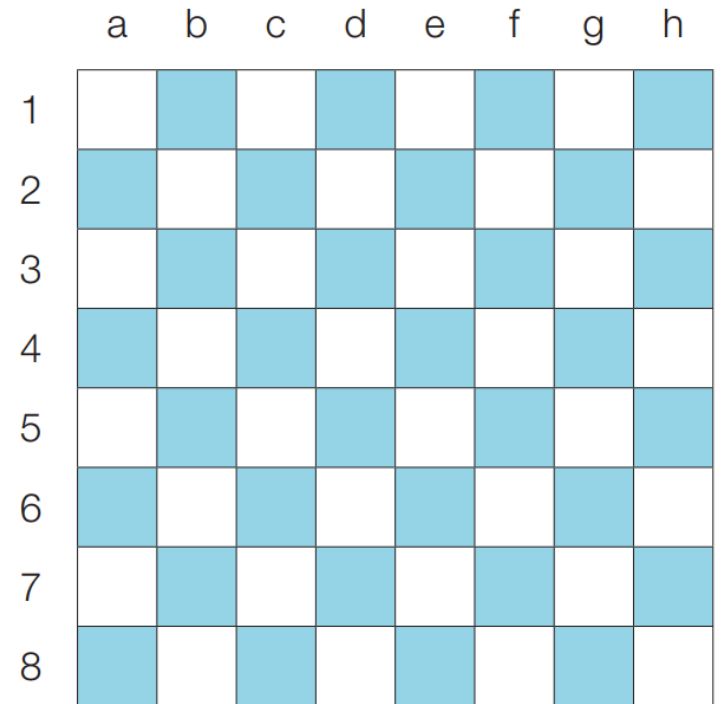


<문제>시각 : 답안 예시(Java)



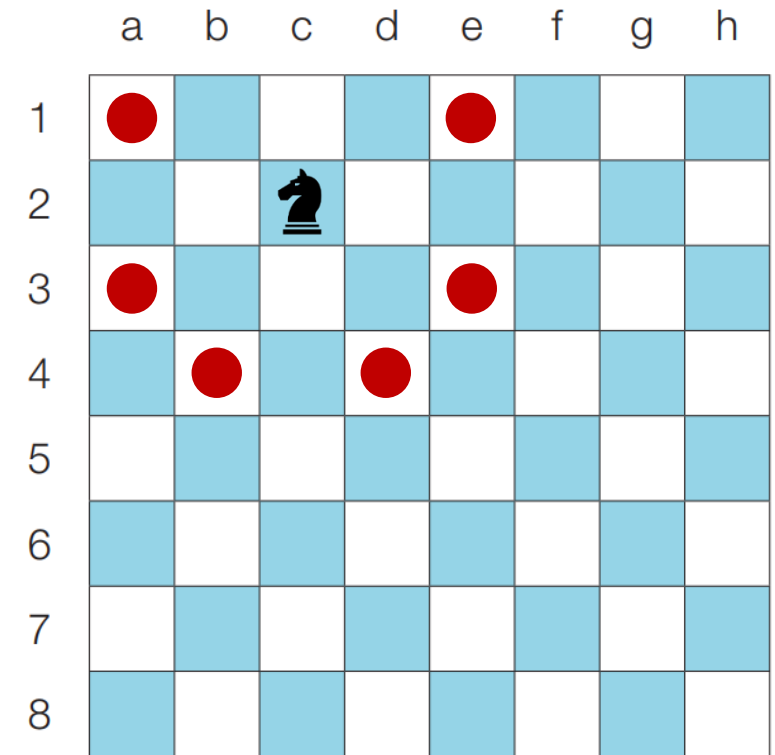
<문제>왕실의 나이트 : 문제 설명

- 행복 왕국의 왕실 정원은 체스판과 같은 8×8 좌표 평면입니다. 왕실 정원의 특정한 한 칸에 나이트가 서 있습니다. 나이트는 매우 충성스러운 신하로서 매일 무술을 연마합니다.
- 나이트는 말을 타고 있기 때문에 이동을 할 때는 L자 형태로만 이동할 수 있으며 정원 밖으로 는 나갈 수 없습니다.
- 나이트는 특정 위치에서 다음과 같은 2가지 경우로 이동할 수 있
1. 수평으로 두 칸 이동한 뒤에 수직으로 한 칸 이동하기
2. 수직으로 두 칸 이동한 뒤에 수평으로 한 칸 이동하기



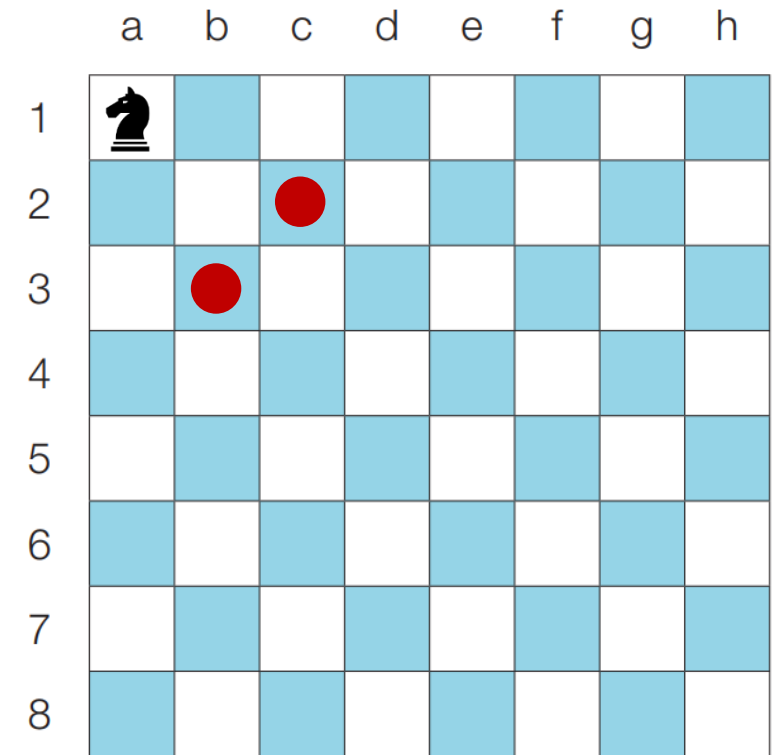
<문제>왕실의 나이트 : 문제 설명

- 이처럼 8×8 좌표 평면상에서 나이트의 위치가 주어졌을 때 나이트가 이동할 수 있는 경우의 수를 출력하는 프로그램을 작성하세요. 왕실의 정원에서 행 위치를 표현할 때는 1부터 8로 표현하며, 열 위치를 표현할 때는 a부터 h로 표현합니다.
 - c2에 있을 때 이동할 수 있는 경우의 수는 6가지입니다



<문제>왕실의 나이트 : 문제 설명

- 이처럼 8×8 좌표 평면상에서 나이트의 위치가 주어졌을 때 나이트가 이동할 수 있는 경우의 수를 출력하는 프로그램을 작성하세요. 왕실의 정원에서 행 위치를 표현할 때는 1부터 8로 표현하며, 열 위치를 표현할 때는 a부터 h로 표현합니다.
 - a1에 있을 때 이동할 수 있는 경우의 수는 2가지입니다



<문제>왕실의 나이트 : 문제 조건

난이도 ●○○ | 풀이 시간 20분 | 시간 제한 1초 | 메모리 제한 128MB

입력 조건

- 첫째 줄에 8×8 좌표 평면상에서 현재 나이트가 위치한 곳의 좌표를 나타내는 두 문자로 구성된 문자열이 입력된다. 입력 문자는 a1처럼 열과 행으로 이뤄진다.

출력 조건

- 첫째 줄에 나이트가 이동할 수 있는 경우의 수를 출력하시오.

입력 예시

a1

출력 예시

2

<문제>왕실의 나이트 : 문제 해결 아이디어

- 요구사항대로 충실히 구현하면 되는 문제입니다.
- 나이트의 8가지 경로를 하나씩 확인하며 각 위치로 이동이 가능한지 확인합니다.
 - 리스트를 이용하여 8가지 방향에 대한 방향 벡터를 정의합니다.

<문제>왕실의 나이트 : 답안 예시(Python)



<문제>왕실의 나이트 : 답안 예시(C++)



<문제>왕실의 나이트 : 답안 예시(Java)

