문제

크기가 N×N인 도시가 있다. 도시는 1×1크기의 칸으로 나누어져 있다. 도시의 각 칸은 빈 칸, 치킨집, 집 중 하나이다. 도시의 칸은 (r, c)와 같은 형태로 나타내고, r행 c열 또는 위에서부터 r번째 칸, 왼쪽에서부터 c번째 칸을 의미한다. r과 c는 1부터 시작한다.

이 도시에 사는 사람들은 치킨을 매우 좋아한다. 따라서, 사람들은 "**치킨 거리**"라는 말을 주로 사용한다. **치킨 거리**는 집과 가장 가까운 치킨집 사이의 거리이다. 즉, 치킨 거리는 집을 기준으로 정해지며, 각각의 집은 **치킨 거리**를 가지고 있다. **도시의 치킨 거리**는 모든 집의 **치킨 거리**의 합이다.

임의의 두 칸 (r_1, c_1)과 (r_2, c_2) 사이의 거리는 $|r_1-r_2| + |c_1-c_2|$로 구한다.

예를 들어, 아래와 같은 지도를 갖는 도시를 살펴보자.

```
0 2 0 1 0
1 0 1 0 0
0 0 0 0 0
0 0 0 1 1
0 0 0 1 2
```

0은 빈 칸, 1은 집, 2는 치킨집이다.

(2, 1)에 있는 집과 (1, 2)에 있는 치킨집과의 거리는 |2-1| + |1-2| = 2, (5, 5)에 있는 치킨집과의 거리는 |2-5| + |1-5| = 7이다. 따라서, (2, 1)에 있는 집의 치킨 거리는 2이다.

(5, 4)에 있는 집과 (1, 2)에 있는 치킨집과의 거리는 |5-1| + |4-2| = 6, (5, 5)에 있는 치킨집과의 거리는 |5-5| + |4-5| = 1이다. 따라서, (5, 4)에 있는 집의 치킨 거리는 1이다.

이 도시에 있는 치킨집은 모두 같은 프랜차이즈이다. 프렌차이즈 본사에서는 수익을 증가시키기 위해 일부 치킨집을 폐업시키려고 한다. 오랜 연구 끝에 이 도시에서 가장 수익을 많이 낼 수 있는 치킨집의 개수는 최대 M개라는 사실을 알아내었다.

도시에 있는 치킨집 중에서 최대 M개를 고르고, 나머지 치킨집은 모두 폐업시켜야 한다. 어떻게 고르면, **도시의 치킨 거리**가 가장 작게 될지 구하는 프로그램을 작성하시오.

입력

첫째 줄에 N(2 ≤ N ≤ 50)과 M(1 ≤ M ≤ 13)이 주어진다.

둘째 줄부터 N개의 줄에는 도시의 정보가 주어진다. → N×N

도시의 정보는 0, 1, 2로 이루어져 있고, 0은 빈 칸, 1은 집, 2는 치킨집을 의미한다. 집의 개수는 2N개를 넘지 않으며, 적어도 1개는 존재한다. 치킨집의 개수는 M보다 크거나 같고, 13보다 작거나 같다.

출력

첫째 줄에 폐업시키지 않을 치킨집을 최대 M개를 골랐을 때, 도시의 치킨 거리의 최솟값을 출력한다.

예제 입력 1 복사

```
5 3
0 0 1 0 0
0 0 2 0 1
0 1 2 0 0
0 0 1 0 0
0 0 0 0 2
```

(2,3)
(3,2)
→ 폐업 안해도됨 ...

예제 출력 1 복사

```
5
```

예제 입력 2 복사

```
5 2
0 2 0 1 0
1 0 1 0 0
0 0 0 0 0
2 0 0 1 1
2 2 0 1 2
```

5개중 3개 폐업.

예제 출력 2 복사

```
10
```

예제 입력 3 복사

```
5 1
1 2 0 0 0
1 2 0 0 0
1 2 0 0 0
1 2 0 0 0
1 2 0 0 0
```

하나빼고 다 모여야.

예제 출력 3 복사

```
11
```

예제 입력 4 복사

```
5 1
1 2 0 2 1
1 2 0 2 1
1 2 0 2 1
1 2 0 2 1
1 2 0 2 1
```

or

예제 출력 4 복사

```
32
```

도시 치킨 거리의 최솟값 = |집(x,y)| - |치킨집(x,y)|

생각나는 방법: 치킨집의 조합을 만들어 그에 따른 도시 치킨 거리의 최솟값을 비교해나가며 정답을 구한다.

↳ 치킨집 list + [(x1,y1), (x2,y2), ···]

최대 M만큼 조합.

집(x1,y1) - 치킨집(x1,y1) = n ↔ 비교

집(x1,y1) - 치킨집(x2,y2) = n

나머지 치킨조합 만큼 반복

⋮

최소거리 = n

↳ 집 전부 탐색할 때까지 → min_distance 비교해서 최소 거리를 도출

minDistance += n