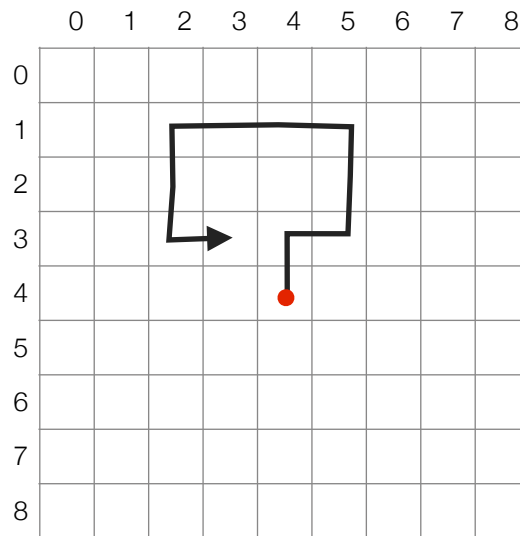


## 실습과제 08

(각 문제에 대해서 주어진 테스트 데이터를 모두 통과하지 못한 경우에는 반드시 그 사실을 명시해야 한다.)

1. [Puppy's Escape] 강아지가  $N \times N$  크기의 2차원 배열의 가운데 위치 ( $N/2, N/2$ )에서 출발한다.  $N$ 은 홀수이다. 상,하,좌,우 4방향으로 인접한 셀(cell)들 중에서 방문한 적이 없는 한 셀을 동일한 확률로 랜덤하게 선택하여 한 칸 이동한다. 가령 아래 그림에서 강아지의 현재 위치는 (3,3)이고 이웃한 위치들 중에서 아직 방문하지 않은 위치는 (2,3)과 (4,3)으로 2곳이다. 따라서 두 위치중 하나를 1/2의 확률로 선택하여 이동한다. 배열의 가장자리 셀에 도착하면 탈출에 성공한 것이다. 하지만 아무 곳으로도 이동할 수 없는 상태에 처하면 탈출에 실패한 것이다. 아래 그림의 예에서 만약 위쪽 방향을 선택하여 이동한다면 그 다음에 어떻게 하더라도 탈출에 성공할 수는 없다. 입력으로 하나의 홀수  $N \leq 100$ 을 받아서 강아지가 탈출에 성공할 확률을 시뮬레이션으로 계산하는 프로그램을 작성하라. 실험 횟수는 10,000번으로 하라.



입력 예	출력 (정답이 없고 유사하면 됨)
5	1.0
9	0.966
21	0.644
51	0.117
71	0.033
99	0.005

2. 1번 문제를 3차원 배열에서 다루는 프로그램을 작성하라. 즉  $N \times N \times N$  크기의 3차원 배열의 가운데 위치 ( $N/2, N/2, N/2$ )에서 출발한다.

입력 예	출력 (정답이 없고 유사하면 됨)
5	1.0
9	0.9995
21	0.9911
51	0.905
71	0.8211
99	0.6802

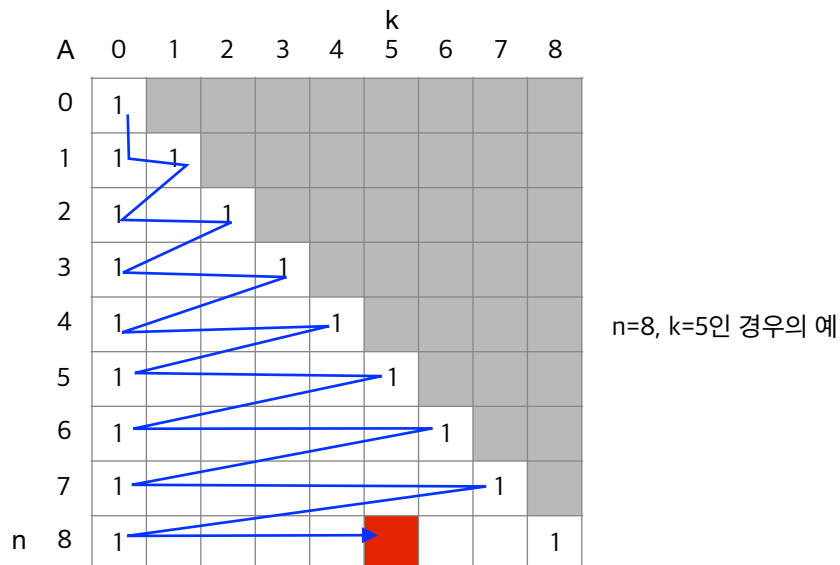
3. 이항계수 (binomial coefficient)  ${}_nC_k$ 는  $n$ 개 중에서  $k$ 개를 선택하는 경우의 수이다.  ${}_nC_k = \frac{n!}{(n-k)!k!}$ 이

지만 이 공식을 이용하여  ${}_nC_k$ 를 계산하는 것은 좋은 방법이 아니다. 왜 그럴까? 일반적으로는 다음의 순환식을 이용하는 것이 더 좋은 방법이다.

$${}_nC_k = 1, k = 0 \text{ 혹은 } n = k \text{인 경우}$$

$${}_nC_k = {}_{n-1}C_{k-1} + {}_{n-1}C_k, \text{ 그렇지 않은 경우}$$

이 순환식을 이용하여  ${}_nC_k$ 의 값을 계산하기 위하여 2차원 배열 A를 사용한다. 배열 A에서 A[n][k]가  ${}_nC_k$ 의 값을 계산하여 저장할 자리이다. 배열에서 대각선 위쪽은 n보다 k가 큰 경우이므로 무의미하다.  $k = 0$  혹은  $n = k$ 인 경우에는  ${}_nC_k = 1$ 이므로 배열의 첫 번째 열과 대각선의 값은 1이다. 순환식에 따르면 이 배열의 임의의 값은 자신의 바로 위쪽 값과 왼쪽-위쪽 대각 방향의 값의 합이다. 우리가 알고 싶은 값이 A[8][5]라면 아래의 그림에서 파란 화살표가 표시하는 순서대로 각 셀의 값을 계산해 나가면 된다. 이 순서로 계산하면 어떤 값을 계산할 때 그 값의 위쪽 값과 왼쪽-위쪽 대각 방향의 값이 항상 먼저 계산되어져 있을 것이기 때문이다. 입력으로 두 양의 정수  $n, k, (k \leq n \leq 100)$ 를 받아서 이런 방법으로  ${}_nC_k$ 의 값을 계산하여 출력하는 프로그램을 작성하라.



입력 예	출력
6 3	20
10 4	210
20 10	184756
100 5	75287520
100 10	1591253560
100 50	-938977944 (정수 오버플로우가 난 것임)
32 16	601080390
32 9	28048800

4. 데이터 파일 input4.txt에는 하나의  $N \times N$  행렬이 저장되어 있다. 파일의 첫 줄에는 행렬의 크기  $N$ 이 저장되어 있고, 이어진  $N$ 줄에는 각 줄마다  $N$ 개의 정수가 저장되어 있다. 이 파일을 읽어서 각 행의 평균과 표준편차, 각 열의 평균과 표준편차를 구해서 아래의 예와 같은 형식으로 출력하는 프로그램을 작성하라.

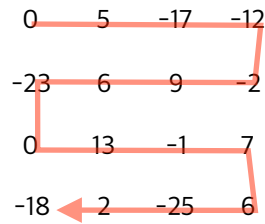
입력 예 (INPUT4.TXT)	출력
4	1 4 19 4 7.000000 7.035624
1 4 19 4	-5 0 8 3 1.500000 4.716991
-5 0 8 3	2 8 2 2 3.500000 2.598076
2 8 2 2	0 2 8 4 3.500000 2.958040
0 2 8 4	-0.500000 3.500000 9.250000 3.250000
	2.692582 2.958040 6.139015 0.829156
8	12 -9 13 7 22 91 0 -81 6.875000 43.861821
12 -9 13 7 22 91 0 -81	9 0 -10 8 14 9 -71 91 6.250000 41.127090
9 0 -10 8 14 9 -71 91	7 44 -18 0 1 12 31 61 17.250000 24.381089
7 44 -18 0 1 12 31 61	0 0 -56 9 12 8 21 22 2.000000 23.232520
0 0 -56 9 12 8 21 22	-9 8 12 42 81 16 9 0 19.875000 26.904635
-9 8 12 42 81 16 9 0	-12 6 8 7 12 21 23 7 9.000000 10.049876
-12 6 8 7 12 21 23 7	7 12 9 22 -7 -9 0 26 7.500000 11.800424
7 12 9 22 -7 -9 0 26	-18 17 0 11 2 8 11 16 5.875000 10.623529
-18 17 0 11 2 8 11 16	-0.5 9.75 -5.25 13.25 17.125 19.5 3.000000 17.750000
	10.428327 14.956186 21.787324 12.285662 25.585335
	28.217902 29.795134 46.780739

5. 데이터 파일 input5.txt에는 하나의  $N \times N$  행렬이 저장되어 있다. 파일의 첫 줄에는 행렬의 크기  $N \leq 100$ 이 저장되어 있고, 이어진  $N$ 줄에는 각 줄마다  $N$ 개의 정수가 저장되어 있다. 이 행렬에서 각 원소의 합이 최대가 되는 부분 행렬을 찾아서 그 최대 합을 출력하는 프로그램을 작성하라. 예를 들어 다음의 행렬에서 합이 최대가 되는 부분 행렬은 사각형으로 둘러싼 부분이고 그 합은 32이다. 부분 행렬이 반드시 정방행렬일 필요는 없다.

	0	5	-17	-12
-23	6	9	-2	
0	13	-1	7	
-18	2	-25	6	

입력 예 (INPUT5.TXT)	출력
4	
1 4 19 4	
-5 0 8 3	64
2 8 2 2	
0 2 8 4	
8	
12 -9 13 7 22 91 0 -81	
9 0 -10 8 14 9 -71 91	
7 44 -18 0 1 12 31 61	
0 0 -56 9 12 8 21 22	601
-9 8 12 42 81 16 9 0	
-12 6 8 7 12 21 23 7	
7 12 9 22 -7 -9 0 26	
-18 17 0 11 2 8 11 16	
4	
0 5 -17 -12	
-23 6 9 -2	32
0 13 -1 7	
-18 2 -25 6	

6. 데이터 파일 input6.txt에는 하나의  $N \times N$  행렬이 저장되어 있다. 파일의 첫 줄에는 행렬의 크기  $N \leq 100$ 이 저장되어 있고, 이어진  $N$ 줄에는 각 줄마다  $N$ 개의 정수가 저장되어 있다. 이 행렬의 원소들을 아래 그림의 화살표처럼 zigzag 순서대로 출력하는 프로그램을 작성하라.



출력순서: 0 5 -17 -12 -2 9 6 -23 0 13 -1 7 6 -25 2 -18

입력 예 (INPUT6.TXT)	출력
3 1 4 19 -5 0 8 2 8 2	1 4 19 8 0 -5 2 8 2
4 0 5 -17 -12 -23 6 9 -2 0 13 -1 7 -18 2 -25 6	0 5 -17 -12 -2 9 6 -23 0 13 -1 7 6 -25 2 -18