/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

试题编号 66

试题名称 期末测试2-题目3-Z字形扫描

时间限制: 1 秒

内存限制: 256MB

问题描述

问题描述

在图像编码的算法中，需要将一个给定的方形矩阵进行Z字形扫描(Zigzag Scan)。给定一个n×n的矩阵，Z字形扫描的过程如下图所示。

对于下面给出的4×4的矩阵，

　　1 5 3 9

　　3 7 5 6

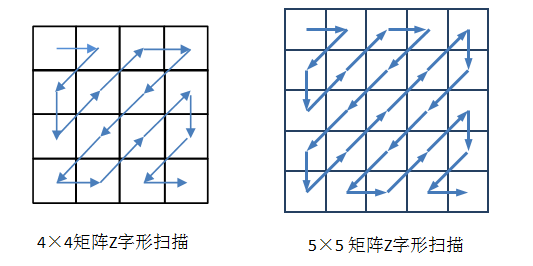
　　9 4 6 4

　　7 3 1 3

　　对其进行Z字形扫描后得到长度为16的序列：

　　1 5 3 9 7 3 9 5 4 7 3 6 6 4 1 3

　　请实现一个Z字形扫描的程序，给定一个n×n的矩阵，输出对这个矩阵进行Z字形扫描的结果。



输入说明

输入的第一行包含一个整数n，表示矩阵的大小。

　　输入的第二行到第n+1行每行包含n个正整数，由空格分隔，表示给定的矩阵。

1≤n≤100，矩阵元素为不超过1000的正整数

输出说明

输出一行，包含n×n个整数，由空格分隔，表示输入的矩阵经过Z字形扫描后的结果。

输入样例

4

1 5 3 9

3 7 5 6

9 4 6 4

7 3 1 3

输出样例

1 5 3 9 7 3 9 5 4 7 3 6 6 4 1 3

提示信息

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

// 去下行注释则禁用 assert()

// #define NDEBUG

#include <assert.h>

int main()

{

int n, a[100][100];

//int n = 4, a[4][4] = { {1,5,3,9},{3,7,5,6},{9,4,6,4},{7,3,1,3} };

// Finite State Machine

// last,next = 1,水平(左-->右)，2竖直(上-->下)，3斜下(右上-->左下)，4斜上(左下-->右上)

int i = 0, j = 0, last = 1, next = 1;

scanf("%d",&n);

for(i = 0; i < n; i++)

for(j = 0; j < n; j++)

scanf("%d",&a[i][j]);

i = 0; j = 0;

printf("%d ", a[i][j]);

while (1)

{

if (i == n - 1 && j == n - 1) break;

last = next;

switch (last)

{

case 1: // 水平

j++;

if (i == 0) next = 3; // 转斜下

else if (i == n - 1) next = 4; // 转斜上

break;

case 2: // 竖直

i++;

if (j == 0) next = 4; // 转斜上

else if (j == n - 1) next = 3; // 转斜下

break;

case 3: // 斜下

i++; j--;

if (i > n - 1) i = n - 1;

if (j < 0) j = 0;

if (j == 0)

{

if (i == n - 1) next = 1; // 转水平

if (i < n - 1) next = 2; // 转竖直

}

else if (i == n - 1) next = 1; // 转水平

break;

case 4: // 斜上

i--; j++;

if (i < 0) i = 0;

if (j > n - 1) j = n - 1;

if (i == 0)

{

if (j == n - 1) next = 2; // 转竖直

if (j < n - 1) next = 1; // 转水平

}

else if(j == n-1) next = 2; // 转竖直

break;

}

printf("%d ", a[i][j]); // 输出last状态下的a[i][j]

}

return 0;

}

int main1()

{

//int n, a[100][100];

int n = 4, a[4][4] = { {1,5,3,9},{3,7,5,6},{9,4,6,4},{7,3,1,3} };

// Finite State Machine

// last,next = 1,水平(左-->右)，2竖直(上-->下)，3斜下(右上-->左下)，4斜上(左下-->右上)

int i = 0, j = 0, last = 1, next = 1;

printf("%d ", a[0][0]);

while (1)

{

if (i == n - 1 && j == n - 1) break;

last = next;

switch (last)

{

case 1: // 水平

assert(i == 0 || i == n - 1);

assert(j < n - 1);

j++;

if (i == 0) next = 3; // 转斜下

else if (i == n - 1) next = 4; // 转斜上

break;

case 2: // 竖直

assert(j == 0 || j == n - 1);

assert(i < n - 1);

i++;

if (j == 0) next = 4; // 转斜上

else if (j == n - 1) next = 3; // 转斜下

break;

case 3: // 斜下

i++; j--;

if (i > n - 1) i = n - 1;

if (j < 0) j = 0;

if (j == 0)

{

if (i == n - 1) next = 1; // 转水平

if (i < n - 1) next = 2; // 转竖直

}

else if (i == n - 1) next = 1; // 转水平

break;

case 4: // 斜上

i--; j++;

if (i < 0) i = 0;

if (j > n - 1) j = n - 1;

if (i == 0)

{

if (j == n - 1) next = 2; // 转竖直

if (j < n - 1) next = 1; // 转水平

}

else if(j == n-1) next = 2; // 转竖直

break;

}

printf("%d ", a[i][j]); // 输出last状态下的a[i][j]

}

return 0;

}

/\* 201412-2 Z字形扫描 \*/

#include <iostream>

using namespace std;

const int EAST = 0;

const int SOUTH = 1;

const int SOUTHWEST = 2;

const int NORTHEAST = 3;

struct {

int drow;

int dcol;

} direct[] = {{0, 1}, {1, 0}, {1, -1}, {-1, 1}};

const int N = 500;

int a[N][N];

int main()

{

int n;

// 输入数据

cin >> n;

for(int i=0; i<n; i++)

for(int j=0; j<n; j++)

cin >> a[i][j];

// Z字形扫描，输出结果

int row=0, col=0, next=EAST;

cout << a[row][col];

while(row != n - 1 || col != n - 1) {

row += direct[next].drow;

col += direct[next].dcol;

cout << " " << a[row][col];

if(next == EAST && row == 0)

next = SOUTHWEST;

else if(next == EAST && row == n - 1)

next = NORTHEAST;

else if(next == SOUTH && col == 0)

next = NORTHEAST;

else if(next == SOUTH && col == n - 1)

next = SOUTHWEST;

else if(next == SOUTHWEST && row == n - 1)

next = EAST;

else if(next == SOUTHWEST && col == 0)

next = SOUTH;

else if(next == NORTHEAST && col == n - 1)

next = SOUTH;

else if(next == NORTHEAST && row == 0)

next = EAST;

}

cout << endl;

return 0;

}