JooHyun – Lee (comkiwer)

TS쏟아진블록

Hancom Education Co. Ltd.

Problem

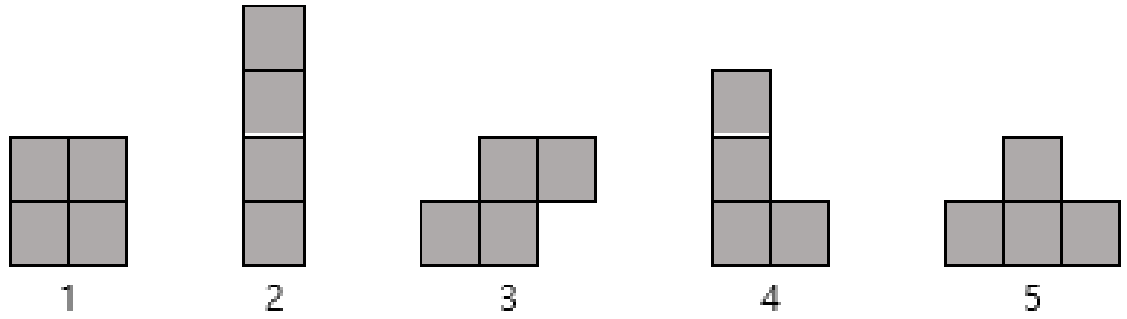
Problem

TS쏟아진블록

테트로미노(Tetromino)는 4개의 칸으로 이루어진 블록을 의미한다.

회전과 반전을 고려하면 서로 다른 종류의 테트로미노 블록은 총 5개가 존재한다.
([Fig. 1] 참조)

회전은 90도, 180도, 270도 회전한 경우만 고려하고 반전은 블록이 뒤집힌 경우를 뜻한다.

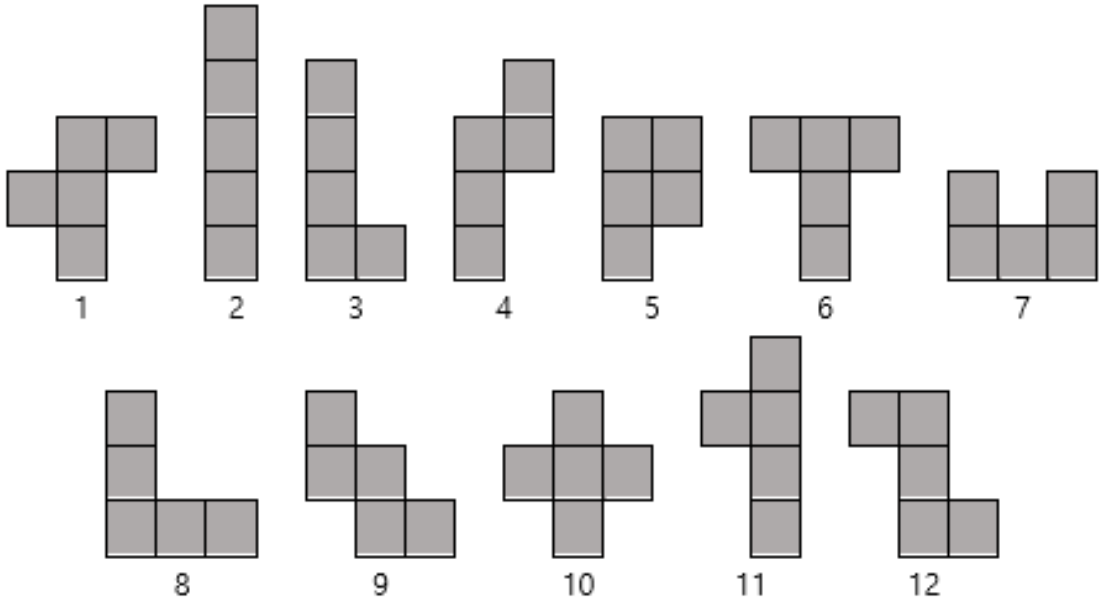


[Fig. 1]

[Fig. 1]에서 블록 아래에 있는 수는 종류를 구별하기 위한 테트로미노 블록 번호이다.

Problem

펜토미노(Pentomino)는 5개의 칸으로 이루어진 블록을 뜻한다.
회전과 반전을 고려하면 서로 다른 종류의 펜토미노 블록은 총 12개가 존재한다. ([Fig. 2] 참조)



[Fig. 2]

[Fig. 2]에서 블록 아래에 있는 수는 종류를 구별하기 위한 펜토미노 블록 번호이다.
N x N 크기의 격자판에 테트로미노 블록들과 펜토미노 블록들이 서로 겹치지 않게 쏟아졌다.
격자판을 보고 각 종류의 블록이 몇 개가 있는지 구해라.
아래 API 설명을 참조하여 각 함수를 구현하라.

Problem

TS쏟아진블록

void countBlock(int N, int mBoard[][], int mTetromino[], int mPentomino[])

격자판에 있는 테트로미노 블록들과 펜토미노 블록들을 종류별로 구별한 후
그 개수를 `mTetromino` 배열과 `mPentomino` 배열에 각각 저장하여 반환한다.

`N`은 격자판의 한 변의 길이를 의미한다. 격자판은 정사각형 모양을 가진다.

`mBoard[r][c]`은 격자판에서 아래 방향으로 `r`번째이고 오른쪽 방향으로 `c`번째인 칸에 있는 블록의 색이다. `r`과 `c`는 `0`부터 시작한다. `(0 ≤ r, c ≤ N - 1)`

블록의 색은 `1` 이상 `9` 이하의 정수로 주어진다. 만약 블록이 없는 빈 칸이면 `0`으로 표현된다.
같은 블록에 속한 칸은 모두 같은 색이다. 서로 이웃하는 블록들은 다른 색으로 구분된다.

void countBlock(int N, int mBoard[][], int mTetromino[], int mPentomino[])

(계속)

격자판에는 테트로미노 블록과 펜토미노 블록 외 다른 블록은 없다.

반환할 때, 블록 번호가 i인 테트로미노 블록의 개수를 mTetromino[i - 1]에 저장하고 블록 번호가 j인 펜토미노 블록의 개수를 mPentomino[j - 1]에 저장한다.

(1 ≤ i ≤ 5, 1 ≤ j ≤ 12)

Parameters

N : 격자판의 한 변의 길이 (5 ≤ N ≤ 1,000)

mBoard[][] : 격자판 상태 (0 ≤ mBoard[r][c] ≤ 9, 0 ≤ r, c ≤ N - 1)

mTetromino[] : 종류별 테트로미노 블록의 개수를 저장하고 반환해야 할 배열

mPentomino[] : 종류별 펜토미노 블록의 개수를 저장하고 반환해야 할 배열

Problem

[제약사항]

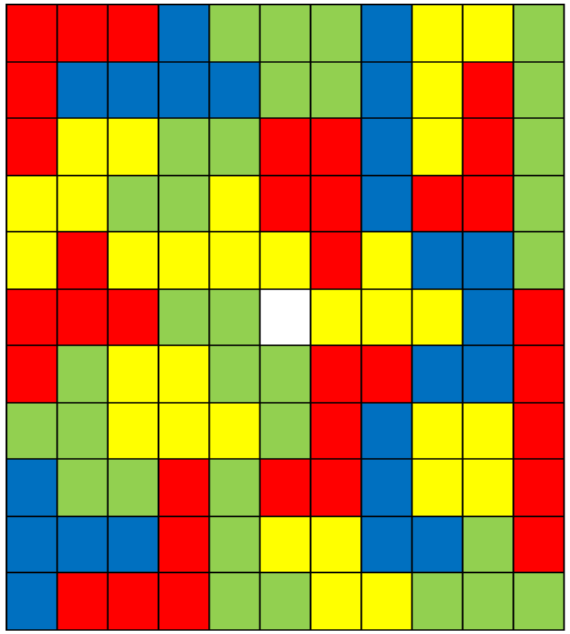
1. **격자판의 한번의 길이** N**은** 5 **이상** 1,000 **이하의 정수이다.**
2. **각 테스트 케이스에서** countBlock() **함수는** 1**번 호출된다.**

Problem analysis

Problem analysis : 예제

격자판의 한 변의 길이가 `N = 11`**인 예를 살펴보자.**
`mBoard` **배열이 다음과 같이 주어졌다고 생각하자.**

```
mBoard[][] =
11143334223
14444334213
12233114213
22332114113
21222212443
11133022241
13223311441
33222314221
43313114221
44413224431
41113322333
```



[Fig. 3]

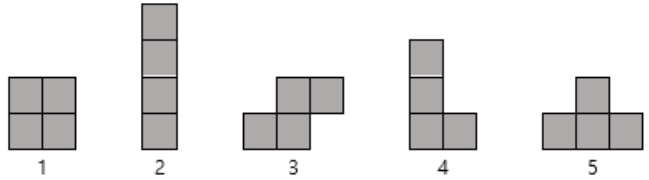
이를 실제 그림으로 표현하면 `[Fig. 3]`**과 같다.**

Problem analysis : 예제

테트로미노 블록 1, 2가 각각 1개 있다.

테트로미노 블록 3, 5가 각각 2개 있다.

테트로미노 블록 4가 4개 있다.



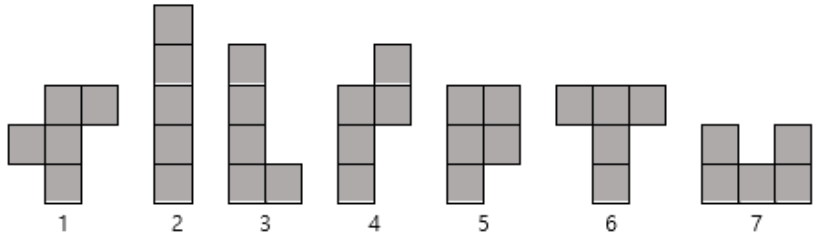
[Fig. 1]

펜토미노 블록 1, 2, 8, 9, 11가 각각 2개 있다.

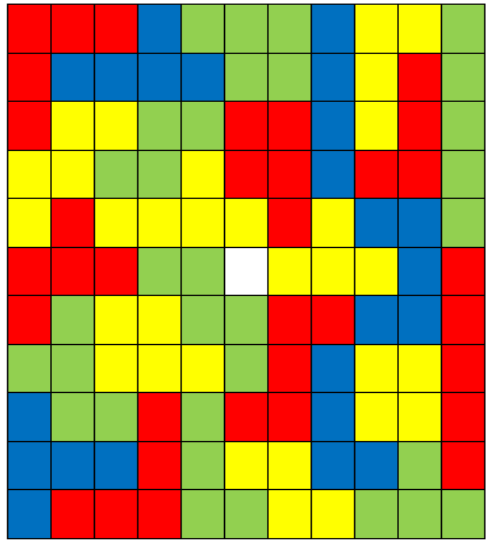
펜토미노 블록 5가 3개 있다.

펜토미노 블록 6, 7, 12은 각각 1개 있다.

펜토미노 블록 3, 4, 10은 없다



[Fig. 2]



[Fig. 3]

그러므로

mTetromino[] = {1, 1, 2, 4, 2},

Pentomino[] = {2, 2, 0, 0, 3, 1, 1, 2, 2, 0, 2, 1} 를 저장하고 반환한다.

Problem analysis : 제약조건 및 API

- 하나의 TC에서 countBlock() 함수는 단 1번 호출된다.
- 격자판의 최대 크기는 1,000 * 1,000 이다.
  이때 블록의 최대 수는 250,000 이다.
- 등장하는 도형은 테트로미노 5가지 펜토미노 12가지이다.
- 하나의 도형은 회전을 고려하면 5*5 크기의 테이블로 표현할 수 있다.

Problem analysis : 제약조건 및 API

- 기본도형 17가지를 어떻게 표현 할 수 있을까?

- countBlock() 함수의 매개변수로 주어지는 mBoard[1000][1000]에서
  기본도형을 어떻게 추출할 수 있을까?

- mBoard[1000][1000]에서 기본도형을 추출한 경우
  기본도형에서 어떻게 찾을 것인가?

Solution sketch

Solution sketch

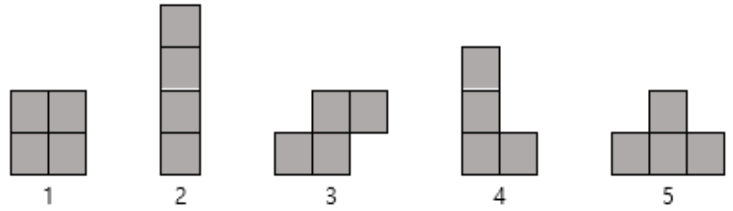
- 하나의 기본도형을 5*5 크기로 표현하자.
- 테트로미노 5개를 0~4번으로
- 펜토미노 12개를 5~16번으로 번호를 부여하자.
- 예를 들어 0번 도형은
  block[0][5][5]에 {{1,1}, {1, 1}, {}, {},{}} 로 저장된다.

1	**1**	0	0	0
1	**1**	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

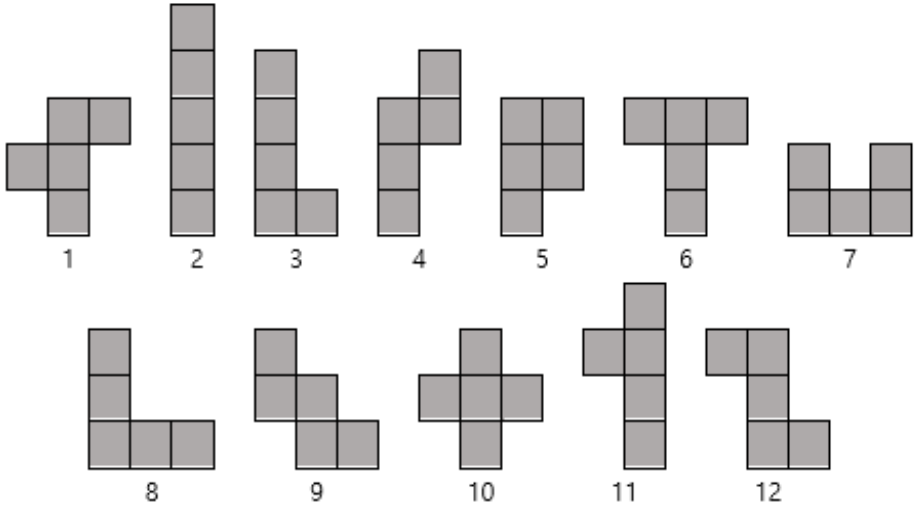
Solution sketch

- 이렇게 모든 도형을 표시하면 다음과 같다.

```
int block[17][5][5] = {
{{1, 1},{1, 1},{},{},{}},
{{1},{1},{1},{1},{}},
{{0, 1, 1},{1, 1},{},{},{}},
{{1},{1},{1,1},{},{}},
{{0,1},{1,1,1},{},{},{}},
{{0,1,1},{1,1},{0,1},{},{}},
{{1},{1},{1},{1},{1}},
{{1},{1},{1},{1,1},{}},
{{0,1},{1,1},{1,},{1},{}},
{{1,1},{1,1},{1},{},{}},
{{1,1,1},{0,1},{0,1},{},{}},
{{1,0,1},{1,1,1},{},{},{}},
{{1},{1},{1,1,1},{},{}},
{{1},{1,1},{0,1,1},{},{}},
{{0,1},{1,1,1},{0,1},{},{}},
{{0,1},{1,1},{0,1},{0,1},{}},
{{1,1},{0,1},{0,1,1},{},{}},
};
```



[Fig. 1]



[Fig. 2]

Solution sketch

- 이제 mBoard[][]에서 도형을 추출해보자.
- 도형이 추출되면 mBoard[][]에는 0을 채워 지우자.
- mBoard[][]의 (0,0) 부터 (N-1, N-1)까지 순회하며
  0인 아닌 수 color를 만난 경우를 생각해 보자.
  - ✓ len = 0 ; // 담긴 도형의 개수
  - ✓ sr = sc = N; // 행번호 최소, 열번호 최소
  - ✓ dfs(또는 bfs)탐색을 진행하여 color와 같은 셀 번호를 row[], col[]에 담는다.
    이때, sr, sc를 업데이트한다. 또한 mBoard[][]에서 셀 내용을 지운다.
  - ✓ 탐색이 끝나면 도형을 shape[5][5]에 그린다.
  - ✓ 이도형을 회전 및 뒤집어가며 기본도형 17개와 비교하여 찾는다.

Solution sketch

- 도형회전과 뒤집기를 위하여 다음 함수들을 사용할 수 있다.

```
int maxRow(int brr[][5]) { // 도형을 구성하는 최대 행번호 구하기
    int i, j, mxr = 0;
    for (i = 0; i < 5; ++i)
        for (j = 0; j < 5; ++j)
            if (brr[i][j]) mxr = i;
    return mxr;
}

void flipVertical(int brr[][5]) { // 상하 뒤집기
    int i, j, k, ed = maxRow(brr);
    for (j = 0; j < 5; ++j) {
        for (i = 0, k = ed; i < k; ++i, --k)
            swap(brr[i][j], brr[k][j]);
    }
}

void rotateCCW90(int brr[][5]) {// 시계방향 90도 회전
    int ed = maxRow(brr), i, j, trr[5][5] = { 0 };
    for (i = 0; i < 5; ++i) {
        for (j = 0; j <= ed; ++j) trr[i][j] = brr[ed - j][i];
    }
    memcpy(brr, trr, sizeof(trr));
}
```

Solution sketch

- 시작위치가 주어졌을 때, 도형을 추출하는 한 예는 다음과 같다.

```
// B[][]는 mBoard[][]이다.
int len, row[5], col[5], sr, sc, color;
void getShape(int r, int c) {
    if (r < 0 || r >= N || c < 0 || c >= N || B[r][c] != color) return;
    B[r][c] = 0, row[len] = r, col[len++] = c;
    sr = min(sr, r), sc = min(sc, c);
    for (int i = 0; i < 4; ++i)
        getShape(r + dr[i], c + dc[i]);
}
```

Solution sketch

- 추출된 도형을 기본도형 그룹 block[17][5][5]에서 찾는 예는 다음과 같다.

```
int Find() {
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 17; ++j)
            if (memcmp(block[j], shape, sizeof(shape)) == 0)
                return j;
        rotateCCW90(shape);
    }
    flipVertical(shape);
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 17; ++j)
            if (memcmp(block[j], shape, sizeof(shape)) == 0)
                return j;
        rotateCCW90(shape);
    }
    return 0;
}
```

Solution sketch

- 이제 countBlock() 구현의 예를 보자.

```
// int shape[5][5]는 전역변수로 선언한다.

void countBlock(int N, int board[MAXN][MAXN], int tetro[5], int pento[12])
{
    ::N = N, B = board;
    memset(cnt, 0, sizeof(cnt));
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j)if (color = B[i][j]) {
            sr = sc = N, len = 0;
            getShape(i, j);
            memset(shape, 0, sizeof(shape));
            for (int k = 0; k < len; ++k)
                shape[row[k] - sr][col[k] - sc] = 1;
            int num = Find();
            cnt[num] ++;
        }
    }

    for (int i = 0; i < 5; ++i) tetro[i] = cnt[i];
    for (int i = 5; i < 17; ++i) pento[i - 5] = cnt[i];
}
```

Solution sketch

[Summary]

- look up 테이블에 도형을 표현할 수 있다.
- 문제의 도형을 회전 및 뒤집기 할 수 있다. (종종 나옴)
- 배열에서 원하는 부분을 추출하여 비교할 수 있다.
- 속도 등 성능최적화를 사용하지 않아도 합격한 분이 많이 나온 문제이다.
- 정확하고 빠른 구현의 완성을 연습하는데 좋은 예제로 볼 수 있다.

[One more step]

- 성능향상을 위하여 hash를 이용해 볼 수 있다.
- 기본도형을 회전해 가면서 나올 수 있는 모든 가능한 경우를 해시 테이블에 등록하여 추출된 도형이 어느 도형인지 빠르게 판정할 수 있다.
- 자세한 구현은 예제코드2를 통하여 알 수 있다.

Code example1

Code example1

TS쏟아진블록

```
#include <cstring>
#include <algorithm>
using namespace std;

#define MAXN 1000

int block[17][5][5] = {
{{1, 1},{1, 1},{},{},{}},
{{1},{1},{1},{1},{}},
{{0, 1, 1},{1, 1},{},{},{}},
{{1},{1},{1,1},{},{}},
{{0,1},{1,1,1},{},{},{}},
{{0,1,1},{1,1},{0,1},{},{}},
{{1},{1},{1},{1},{1}},
{{1},{1},{1},{1,1},{}},
{{0,1},{1,1},{1,},{1},{}},
{{1,1},{1,1},{1},{},{}},
{{1,1,1},{0,1},{0,1},{},{}},
{{1,0,1},{1,1,1},{},{},{}},
{{1},{1},{1,1,1},{},{}},
{{1},{1,1},{0,1,1},{},{}},
{{0,1},{1,1,1},{0,1},{},{}},
{{0,1},{1,1},{0,1},{0,1},{}},
{{1,1},{0,1},{0,1,1},{},{}},
};
```

Code example1

TS쏟아진블록

```
int shape[5][5];
int cnt[20], flag = 1;
int N, (*B)[MAXN];
int dr[] = { -1, 1, 0, 0 }, dc[] = { 0, 0, -1, 1 };

int maxRow(int brr[][5]) { // 도형을 구성하는 최대 행번호 구하기
    int i, j, mxr = 0;
    for (i = 0; i < 5; ++i)
        for (j = 0; j < 5; ++j)
            if (brr[i][j]) mxr = i;
    return mxr;
}

void flipVertical(int brr[][5]) { // 상하 뒤집기
    int i, j, k, ed = maxRow(brr);
    for (j = 0; j < 5; ++j) {
        for (i = 0, k = ed; i < k; ++i, --k)
            swap(brr[i][j], brr[k][j]);
    }
}
```

Code example1

TS쏟아진블록

```
void rotateCCW90(int brr[][5]) {// 시계방향 90도 회전
    int ed = maxRow(brr), i, j, trr[5][5] = { 0 };
    for (i = 0; i < 5; ++i) {
        for (j = 0; j <= ed; ++j) trr[i][j] = brr[ed - j][i];
    }
    memcpy(brr, trr, sizeof(trr));
}

int len, row[5], col[5], sr, sc, color;
void getShape(int r, int c) {
    if (r < 0 || r >= N || c < 0 || c >= N || B[r][c] != color) return;
    B[r][c] = 0, row[len] = r, col[len++] = c;
    sr = min(sr, r), sc = min(sc, c);
    for (int i = 0; i < 4; ++i)
        getShape(r + dr[i], c + dc[i]);
}
```

Code example1

TS쏟아진블록

```
int Find() {
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 17; ++j)
            if (memcmp(block[j], shape, sizeof(shape)) == 0)
                return j;
        rotateCCW90(shape);
    }
    flipVertical(shape);
    for (int i = 0; i < 4; ++i) {
        for (int j = 0; j < 17; ++j)
            if (memcmp(block[j], shape, sizeof(shape)) == 0)
                return j;
        rotateCCW90(shape);
    }
    return 0;
}
```

Code example1

TS쏟아진블록

```
void countBlock(int N, int board[MAXN][MAXN], int tetro[5], int pento[12])
{
    ::N = N, B = board;
    memset(cnt, 0, sizeof(cnt));
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j)if (color = B[i][j]) {
            sr = sc = N, len = 0;
            getShape(i, j);
            memset(shape, 0, sizeof(shape));
            for (int k = 0; k < len; ++k)
                shape[row[k] - sr][col[k] - sc] = 1;
            int num = Find();
            cnt[num] ++;
        }
    }

    for (int i = 0; i < 5; ++i) tetro[i] = cnt[i];
    for (int i = 5; i < 17; ++i) pento[i - 5] = cnt[i];
}
```

Code example2

Code example2

TS쏟아진블록

```
#include <cstring>
#include <algorithm>
#include <unordered_map>
using namespace std;
#define MAXN 1000

int block[17][5][5] = {
{{1, 1},{1, 1},{},{},{}},
{{1},{1},{1},{1},{}},
{{0, 1, 1},{1, 1},{},{},{}},
{{1},{1},{1,1},{},{}},
{{0,1},{1,1,1},{},{},{}},
{{0,1,1},{1,1},{0,1},{},{}},
{{1},{1},{1},{1},{1}},
{{1},{1},{1},{1,1},{}},
{{0,1},{1,1},{1,},{1},{}},
{{1,1},{1,1},{1},{},{}},
{{1,1,1},{0,1},{0,1},{},{}},
{{1,0,1},{1,1,1},{},{},{}},
{{1},{1},{1,1,1},{},{}},
{{1},{1,1},{0,1,1},{},{}},
{{0,1},{1,1,1},{0,1},{},{}},
{{0,1},{1,1},{0,1},{0,1},{}},
{{1,1},{0,1},{0,1,1},{},{}},
};
```

Code example2

TS쏟아진블록

```
int W[5][5] = { {   1,    2,    4,   8, 1}, \
                {  16,   32,   64, 128}, \
                { 256,  512, 1024}, \
                {2048, 4096},
                {1} };
unordered_map<int, int> htab;
int cnt[20], flag = 1;
int N, (*B)[MAXN];
int dr[] = { -1, 1, 0, 0 }, dc[] = { 0, 0, -1, 1 };

int maxRow(int brr[][5]) { // 도형을 구성하는 최대 행번호 구하기
    int i, j, mxr = 0;
    for (i = 0; i < 5; ++i)
        for (j = 0; j < 5; ++j)
            if (brr[i][j]) mxr = i;
    return mxr;
}

void flip(int brr[][5]) { // 상하 뒤집기
    int i, j, k, ed = maxRow(brr);
    for (j = 0; j < 5; ++j) {
        for (i = 0, k = ed; i < k; ++i, --k)
            swap(brr[i][j], brr[k][j]);
    }
}
```

Code example2

TS쏟아진블록

```
void rotate90(int brr[][5]) {// 시계방향 90도 회전
    int ed = maxRow(brr), i, j, trr[5][5] = { 0 };
    for (i = 0; i < 5; ++i) {
        for (j = 0; j <= ed; ++j) trr[i][j] = brr[ed - j][i];
    }
    memcpy(brr, trr, sizeof(trr));
}

int getCode(int block[][5], int code = 0) { // 해시 코드 얻기
    for (int i = 0; i < 5; ++i) {
        for (int j = 0; j < 5; ++j)
            code += block[i][j] * W[i][j];
    }
    return code;
}
```

Code example2

TS쏟아진블록

```
void init() {  // 샘플 도형 회전, 뒤집어 해시 코드 구하고 도형 번호 부여하기
    flag = 0;
    for (int i = 0; i < 17; ++i) {
        for (int j = 0; j < 4; ++j) {
            rotate90(block[i]);
            int code = getCode(block[i]);
            htab[code] = i;
        }
        flip(block[i]);
        for (int j = 0; j < 4; ++j) {
            rotate90(block[i]);
            int code = getCode(block[i]);
            htab[code] = i;
        }
    }
}

int len, row[5], col[5], sr, sc, color;
void trace(int r, int c) {  // 제시된 도형 찾기
    if (r < 0 || r >= N || c < 0 || c >= N || B[r][c] != color) return;
    B[r][c] = 0, row[len] = r, col[len++] = c;
    sr = min(sr, r), sc = min(sc, c);
    for (int i = 0; i < 4; ++i)
        trace(r + dr[i], c + dc[i]);
}
```

Code example2

TS쏟아진블록

```
void countBlock(int N, int board[MAXN][MAXN], int tetro[5], int pento[12])
{
    ::N = N, B = board;
    memset(cnt, 0, sizeof(cnt));
    if (flag)  init();
    for (int i = 0; i < N; ++i) {
        for (int j = 0; j < N; ++j)if (color = B[i][j]) {
            len = 0, sr = sc = N;
            trace(i, j);
            int code = 0;
            for (int k = 0; k < len; ++k) {
                int r = row[k] - sr, c = col[k] - sc;
                code += W[r][c];
            }
            int shape = htab[code];
            cnt[shape] ++;
        }
    }

    for (int i = 0; i < 5; ++i) tetro[i] = cnt[i];
    for (int i = 5; i < 17; ++i) pento[i - 5] = cnt[i];
}
```

Thank you.