2 차원 점

2 차원 평면 상에서 한 점의 위치는 좌표의 원점과 그 점 사이의 x-축과 y-축 방향의 거리로 나타낸다. 어떤 점 p 가 원점으로부터 x-축과 y-축으로 각각 a 와 b 만큼 떨어 졌을 때, p 의 좌표를 (a, b)로 나타내고, 점 p 를 $p(a,b)$로 표시하기도 한다.

2 차원 평면 상에서 두 점 $p(x_1, y_1)$, $q(x_2, y_2)$ 사이의 직선거리(Euclidean distance) $d_2(p,q)$ 와 직각거리(rectilinear distance) $d_1(p,q)$ 는 다음과 같이 정의한다.

$$d_2(p,q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$
$$d_1(p,q) = |x_1 - x_2| + |y_1 - y_2|$$

예를 들어, 두 점 (2, 3), (5, 7) 의 직선거리는 5 이며, 직각거리는 |2-5|+|3-7|=3+4=7 이다.

다음과 같이 2 차원 상의 점를 위한 클래스 MyPoint 를 정의할 때, 클래스 MyPoint 의 각 멤버 함수를 다음과 같은 테스트 프로그램 TestMyPoint.cpp 가 정확하게 동작하도록 구현하시오.

MyPoint.h

```
#ifndef _MY_POINT_H_
#define _MY_POINT_H_

class MyPoint
{
public:
    // constructors
    MyPoint ();
    MyPoint (int coordX, int coordY);
    MyPoint (const MyPoint& p);

    // accessor functions
    int getX() const;
    int getY() const;

    // mutator function
    void setX(int x);
    void setY(int y);

    // comparison operators
    bool operator== (const MyPoint& p) const;
    bool operator!= (const MyPoint& p) const;

    // utility functions
    int dist2sqr(const MyPoint& p) const; // 직선거리(Euclidean distance)의 제곱을
계산
    int dist1(const MyPoint& p) const;    // 직각거리(rectilinear distance)를 계산

private:
    int x, y;
```

```
};

#endif // _MY_POINT_H_
```

MyPoint.cpp

```
#include <cstdlib>
#include "MyPoint.h"

// constructors
MyPoint::MyPoint ()
:x(0), y(0)        // set default to origin (0,0)
{
}

MyPoint::MyPoint (int coordX, int coordY)
:x(coordX), y(coordY)
{
}

MyPoint::MyPoint (const MyPoint& p)
{
}

// accessor functions
int MyPoint::getX () const
{
    return x;
}

int MyPoint::getY () const
{
}

// mutator functions
void MyPoint::setX (int coordX)
{
    x = coordX;
}

void MyPoint::setY (int coordY)
{
}

bool MyPoint::operator== (const MyPoint& p) const
{
}

bool MyPoint::operator!= (const MyPoint& p) const
{
    return !operator==(p);
}

// 직선거리(Euclidean distance)의 제곱을 계산
int MyPoint::dist2sqr(const MyPoint& p) const
{
}
```

```
// 직각거리(rectilinear distance)를 계산
int MyPoint::dist1(const MyPoint& p) const
{
}
```

TestMyPoint.cpp

```
#include <iostream>
#include "MyPoint.h"
using namespace std;

int main()
{
    int numTestCases;

    cin >> numTestCases;

    for(int i=0; i<numTestCases; i++)
    {
        int coordX, coordY;

        cin >> coordX >> coordY;
        MyPoint p1(coordX, coordY);

        cin >> coordX >> coordY;
        MyPoint p2(coordX, coordY);

        if(p1 == p2)
            cout << "1" << " ";
        if(p1 != p2)
            cout << "0" << " ";

        cout << p1.dist2sqr(p2) << " " << p1.dist1(p2) << " ";

        coordX = p1.getX();
        coordY = p1.getY();

        MyPoint p3(p1);

        p3.setX(coordY);
        p3.setY(coordX);

        cout << p1.dist2sqr(p3) << " " << p1.dist1(p3) << endl;
    }

    return 0;
}
```

입력

입력은 표준입력(standard input)을 사용한다. 입력은 t 개의 테스트 케이스로 주어진다. 입력의 첫 번째 줄에 테스트 케이스의 개수를 나타내는 정수 t 가 주어진다. 두 번째 줄부터 t 개의 줄에는 한 줄에 한 개의 테스트 케이스에 해당하는 네 개의 정수 a b c d 가 주어진다. 이 정수는 각

각 두 개의 점의 좌표 $p(a, b)$, $q(c, d)$ 이다. 각 정수들 사이에는 한 개의 공백이 주어지며, 잘못된 데이터가 입력되는 경우는 없다.

출력

출력은 표준출력(standard output)을 사용한다. 입력되는 테스트 케이스의 순서대로 다음 줄에 이어서 각 테스트 케이스의 결과를 출력한다. 각 테스트 케이스에 해당하는 출력의 첫 줄에 먼저 두 점의 위치가 같은 경우에는 1 을 출력하고, 그렇지 않는 경우에는 0 을 출력한다. 그 다음으로 두 점사이의 직선거리의 제곱과 직각거리를 출력한다. 그 다음으로는 첫 번째 점 p 와 점 $r(b,a)$ 사이의 직선거리의 제곱과 직각거리를 출력한다. 각 정수들 사이에는 한 개의 공백을 둔다.

입력과 출력의 예

입력	출력
3 1 2 1 2 4 3 2 1 2 -3 3 7	1 0 0 2 2 0 8 4 2 2 0 101 11 50 10