解法１

ピタゴラスの定理を利用し、 a^2 + b^2 = c^2 を満たすものを探す

```
struct Vector {
    int x,y;
    Vector(int x = 0,int y =0): x(x),y(y){}
    Vector operator-(const Vector &p) const{
        return Vector(x - p.x, y - p.y);
    }
    int dot(const Vector& v) const {
        return x*v.x + y*v.y;
    }
};

int norm2(Vector a, Vector b){ // normはベクトルの長さの概念
    Vector p = b-a;
    return p.x*p.x + p.y*p.y;
}

void solve_admin_1(void){
    Vector a, b, c;
    cin >> a.x >> a.y;
    cin >> b.x >> b.y;
    cin >> c.x >> c.y;
    int A = norm2(b,c);
    int B = norm2(a,c);
    int C = norm2(a,b);
    if(A+B == C || A+C == B || B+C == A) cout << "Yes" << endl;
    else cout << "No" << endl;
    return;

}
```

解法２：ベクトル

ベクトルの内積が０であるかをみる。 Aを頂点として、B,Cのベクトルで考える。 B(xb,yb),C(xc,yc)とすると内積はxb * xc+yb * ycで求めることができる。また、xb = xb-xa,xc = xc-xaで表せる。

```
struct Vector {
    int x,y;
    Vector(int x = 0,int y =0): x(x),y(y){}
    Vector operator-(const Vector &p) const{
        return Vector(x - p.x, y - p.y);
    }
    int dot(const Vector& v) const {
        return x*v.x + y*v.y;
    }
```

```
};

void solve_admin_2(void){
    Vector a, b, c;
    cin >> a.x >> a.y;
    cin >> b.x >> b.y;
    cin >> c.x >> c.y;
    bool ok = false;
    for(int i = 0; i < 3; i++){
        if((b-a).dot(c-a) == 0) ok = true;
        swap(a,b); swap(b,c);
    }

}
```

解説

- const修飾子 以下のようにメンバ関数に対してconstをつけると、メンバ関数を呼び出したときに変更されるの防ぐことができ、保守性を高めることができる[1]。

```
Vector operator-(const Vector &p) const{
    //以下のような変更コードを入れるとコンパイルエラーとなる。
    //x = 3;
    return Vector(x - p.x, y - p.y);
}
```

参考

- Atcoder 362_B