```
//
//  Shape.h
//

#ifndef Shape_h
#define Shape_h

class Shape {
    Shape() {}
    void GetBoundingBox(int &x1, int &y1, int &x2, int &y2) const;
    virtual bool IsIn(int x, int y) const = 0;
private:
    int x1=-1, y1=-1, x2=-1, y2=-1;
}

#endif

//
//  Shape.cpp
//

#include "Shape.h"

void Shape::GetBoundingBox(int &x1, int &y1, int &x2, int &y2)
{
    x1 = this->x1;
    x2 = this->x2;
    y1 = this->y1;
    y2 = this->y2;
}
```

```
//
//  Square.h
//

#include "Shape.h"

#ifndef Shape_h
#define Shape_h

class Square : public Shape {
    Square(int x1, int y1, int width);
    bool IsIn(int x, int y) const;
private:
    int width;
}

#endif

//
//  Square.cpp
//

#include "Square.h"

Square::Square(int x, int y, int width)
{
    this->width = width;
    x1 = x;
    y1 = y;
    x2 = x+width;
    y2 = y+width;
}

bool Square::IsIn(int x, int y) const
{
    return (x >= x1) && (x <= x2) && (y >= y1) && (y <= y2);
}
```

```
//
//  Circle.h
//
//

#ifndef Shape_h
#define Shape_h

#include "Shape.h"

class Circle : public Shape {
    Circle(int x, int y, int radius);
    bool IsIn(int x, int y) const;
private:
    int centerx, centery, radius;
}

#endif


//
//  Circle.cpp
//

#include "Circle.h"

Circle::Circle(int x, int y, int radius)
{
    centerx = x;
    centery = y;
    radius = radius;
}

bool Circle::IsIn(int x, int y) const
{
    return (x-centerx)*(x-centerx)+(y-centery)*(y-centery) <=
                radius*radius;
}
```

```cpp
//
//  main.cpp
//

#include "Shape.h"

int main(int argc, const char * argv[])
{
    Circle c1(0, 0, 10);
    Circle c2(10, 10, 10);
    Circle c3(10, 10, 1);

    if (Overlap(&c1, &c2))
    {
           cout << "C1 and C2 overlap" << endl;
    }
    else {
           cout << "C1 and C2 don't overlap" << endl;
    }

    if (Overlap(&c1, &c3))
    {
           cout << "C1 and C3 overlap" << endl;
    }
    else {
           cout << "C1 and C3 don't overlap" << endl;
    }

    return 0;
}

void Overlap(Shape &s1, Shape &s2)
{
    int x1, x2, y1, y2;
    s1.GetBoundingBox(x1, y1, x2, y2);
    for (int x = x1; x <= x2; x++)
    {
           for (int y = y1; y <= y2; y++)
           {
                 if (s1.IsIn(x, y) && s2.IsIn(x, y))
                       return true;
           }
    }
    return false;
}
```