Educational Codeforces Round #2

Problem D – Area of Two Circle's Intersection

Prof. Edson Alves

Faculdade UnB Gama

Educational Codeforces Round #2 – Problem D: Area of Two Circle's Intersection

Problema

You are given two circles. Find the area of their intersection.

Entrada e saída

Input

The first line contains three integers x_1, y_1, r_1 $(-10^9 \leq x_1, y_1 \leq 10^9, 1 \leq r_1 \leq 10^9)$ - the position of the center and the radius of the first circle.

The second line contains three integers x_2, y_2, r_2 $(-10^9 \leq x_2, y_2 \leq 10^9, 1 \leq r_2 \leq 10^9)$ - the position of the center and the radius of the second circle.

Output

Print the area of the intersection of the circles. The answer will be considered correct if the absolute or relative error doesn't exceed 10^{-6}.

Exemplo de entradas e saídas

Sample Input

```
0 0 4
6 0 4
```

```
0 0 5
11 0 5
```

Sample Output

```
7.25298806364175601379
```

```
0.00000000000000000000
```

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema
- No primeiro caso os círculos não tem interseção, isto é, a distância d entre os pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ é maior ou igual ao dobro da soma dos raios

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema
- No primeiro caso os círculos não tem interseção, isto é, a distância d entre os pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ é maior ou igual ao dobro da soma dos raios
- Assim, a área de interseção será igual a zero

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema
- No primeiro caso os círculos não tem interseção, isto é, a distância d entre os pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ é maior ou igual ao dobro da soma dos raios
- Assim, a área de interseção será igual a zero
- No segundo caso, um círculo contém o outro, isto é, $d \leq r_1 + r2$

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema
- No primeiro caso os círculos não tem interseção, isto é, a distância d entre os pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ é maior ou igual ao dobro da soma dos raios
- Assim, a área de interseção será igual a zero
- No segundo caso, um círculo contém o outro, isto é, $d \leq r_1 + r2$
- Logo a interseção terá a mesma área do círculo de menor raio

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema
- No primeiro caso os círculos não tem interseção, isto é, a distância d entre os pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ é maior ou igual ao dobro da soma dos raios
- Assim, a área de interseção será igual a zero
- No segundo caso, um círculo contém o outro, isto é, $d \leq r_1 + r2$
- Logo a interseção terá a mesma área do círculo de menor raio
- No terceiro e último caso os dois círculos se tocam em exatamente dois pontos

Solução com complexidade $O(1)$

- É preciso tratar 3 casos especiais para a solução correta deste problema
- No primeiro caso os círculos não tem interseção, isto é, a distância d entre os pontos $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ é maior ou igual ao dobro da soma dos raios
- Assim, a área de interseção será igual a zero
- No segundo caso, um círculo contém o outro, isto é, $d \leq r_1 + r2$
- Logo a interseção terá a mesma área do círculo de menor raio
- No terceiro e último caso os dois círculos se tocam em exatamente dois pontos
- A área será a soma dos segmentos definidos pelos triângulos formados pelos centros e dos pontos de interseção

Solução AC com complexidade $O(1)$

```cpp
#include <bits/stdc++.h>

using namespace std;
using ll = long long;

constexpr long double PI = acosl(-1.0);

long double intersection_area(long double r, long double R, long double d)
{
    auto angle = acosl((r*r + d*d - R*R)/(2*r*d));  // Lei dos cossenos
    auto sector = angle * r * r;                    // Setor de 2*angle
    auto T = r * r * cosl(angle) * sinl(angle);     // Aŕea do triângulo

    return sector - T;
}
```

Solução AC com complexidade $O(1)$

```cpp
void solve(ll X1, ll Y1, ll R1, ll X2, ll Y2, ll R2)
{
    // Não se interceptam ou se tocam em um único ponto
    auto dist2 = (X1 - X2)*(X1 - X2) + (Y1 - Y2)*(Y1 - Y2);

    if (dist2 >= (R1 + R2)*(R1 + R2))
    {
        cout << "0.00000000000000000000\n";
        return;
    }

    // O primeiro círculo será o de maior raio
    if (R2 > R1)
    {
        swap(X1, X2);
        swap(Y1, Y2);
        swap(R1, R2);
    }
```

Solução AC com complexidade $O(1)$

```cpp
    // O menor está contido no maior: a resposta é a área do menor
    if (dist2 <= (R1 - R2)*(R1 - R2))
    {
        cout.precision(20);
        cout << PI*R2*R2 << '\n';
        return;
    }

    // Dois pontos de interseção
    auto d = sqrtl(dist2);
    auto A1 = intersection_area(R1, R2, d);
    auto A2 = intersection_area(R2, R1, d);

    cout.precision(20);
    cout << A1 + A2 << endl;
}
```