El circuito tiene dos entradas: X y Y. Según el diagrama:

-

La compuerta **NAND** recibe X y Y.

-
-

La salida del **NAND** va a una compuerta **OR**, junto con X (directo o negado, según el diagrama).

-
-

La salida final es C.

-

Tabla de verdad

X	Y	NAND(X,Y)	OR(NAND, X)	C
0	0	1	1	1
0	1	1	1	1
1	0	1	1	1
1	1	0	1	1

 Explicación rápida:

-

NAND solo da 0 si ambas entradas son 1.

-
-

OR da 1 si al menos una entrada es 1.

-
-

Como X también entra al OR, siempre hay al menos un 1 → entonces C siempre es 1.

-

```
// NAND: devuelve false solo si ambos son true

def nand(x) = def(y) = not(and(x)(y))


// OR: devuelve true si al menos uno es true

def or(x) = def(y) = if x then true else y


// AND: devuelve true solo si ambos son true

def and(x) = def(y) = if x then y else false


// NOT: invierte el valor

def not(p) = if p then false else true

def C(x) = def(y) = or(nand(x)(y))(x)
```

```
#include <iostream>
using namespace std;

// Función NOT
bool NOT(bool p) {
   return !p;
}

// Función AND
bool AND(bool a, bool b) {
   return a && b;
}

// Función NAND
bool NAND(bool a, bool b) {
   return !AND(a, b);
}

// Función OR
bool OR(bool a, bool b) {
   return a || b;
}

// Función C que representa el circuito
bool C(bool x, bool y) {
   return OR(NAND(x, y), x);
}

int main() {
   // Pruebas de todos los casos
   cout << "C(0,0) = " << C(false, false) << endl;
   cout << "C(0,1) = " << C(false, true) << endl;
   cout << "C(1,0) = " << C(true, false) << endl;
   cout << "C(1,1) = " << C(true, true) << endl;

   return 0;
}
```