# Hamming Distance

## 题目：

Hamming Distance指的是两个整数的二进制表示对应位上不一致的数目。 给定两个整数x，y，统计Hamming Distance.  
  0 <= x,y <= 2^32

example

Input: x = 1, y = 4

Output: 2

Explanation:

1 (0 0 0 1)

4 (0 1 0 0)

↑ ↑

The above arrows point to positions where the corresponding bits are different.

## 思路：

1 将两个整数转换为二进制，比较每一个对应位，统计dfferent的个数。 使用bitset结构体用初始化功能，可以直接将整形转换为二进制，遍历统计即可。

2 使用位运算符（<<,>>）和位与(&)，位异或(^)运算符，通过temp = x^y得到异或后的的值，然后再temp&1，检查最低一位是否是1(即不同)，temp>>=1 将二进制位向右边移动，最低位移除，循环直到temp为0。

## 代码：

思路1（知识面广）：

#include<iostream>

#include<bitset>

using namespace std;

class Solution {

public:

int hammingDistance(int x, int y) {

bitset<32> bs1(x);

bitset<32> bs2(y);

int ct=0;

for(int i=0;i<32;++i){

if(bs1[i]!=bs2[i]) ct++;

}

return ct;

}

};

int main()

{

int x=1,y=4;

int result=0;

Solution sol;

result = sol.hammingDistance(x,y);

cout<<result;

}

思路2：（技巧性强）

#include<iostream>

#include<bitset>

using namespace std;

class Solution {

public:

int hammingDistance(int x, int y) {

int ct=0;

int temp = x^y;

while(temp){

int j = temp & 1;

if(j==1) ct++;

temp = temp >> 1;

}

return ct;

}

};

int main()

{

int x=1,y=4;

int result=0;

Solution sol;

result = sol.hammingDistance(x,y);

cout<<result;

}