# Self Dividing Number

## 题目：

A self-dividing number is a number that is divisible by every digit it contains.

For example, 128 is a self-dividing number because 128 % 1 == 0, 128 % 2 == 0, and 128 % 8 == 0.

Also, a self-dividing number is not allowed to contain the digit zero.

Given a lower and upper number bound, output a list of every possible self dividing number, including the bounds if possible.

The boundaries of each input argument are 1 <= left <= right <= 10000.

自整除数是由它所包含的每一个数字整除的数。

例如，128是一个自划分的数字，因为128%＝1＝0, 128＝2＝0，和128%＝8＝0。

另外，不允许自分数字包含零位。

给定一个下界和上界，输出一个可能的自划分数的列表，如果可能的话，包括边界。

每个输入参数的边界是1 <左= < = < = 10000。

example:

Input:

left = 1, right = 22

Output: [1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 15, 22]

## 思路：

取出[right,left]内的每一个数n，在循环用%取出n的每一位，同时检测每一位是否符合条件(digit==0 以及 n%digit!=0 不符合)。

## 代码：

#include<iostream>

#include<vector>

using namespace std;

class Solution

{

public:

vector<int> selfDividingNumbers(int left, int right)

{

vector<int> result;

for(int i=left;i<=right;++i)

{

int temp = i;

int digit;

bool flag = false;

while(temp!=0)

{

digit = temp%10;

if(digit==0)//含有0退出

{

flag = true;

break;

}

if(i%digit!=0)//除不尽退出

{

flag = true;

break;

}

temp /= 10;

}

if(flag==true) continue;

result.push\_back(i);

}

return result;

}

};

int main()

{

Solution sol;

vector<int> a;

a = sol.selfDividingNumbers(22,222);

for(int i=0; i<a.size(); i++)

{

cout<<a[i]<<endl;

}

}