**★☆☆☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：10396: Vampire Numbers**

**解題者：陳昱維**

**解題日期：2021年6月1日**

**題意：**

 兩串同樣長度為n/2的數字相乘後，得到長度為n的積，若相乘的兩數每一位數字組合起來排列後，與他們的積相同，則該數為Vampire Number。但是相乘的兩數結尾不可都是0。題目要求找出所有4位數、6位數和8位數的偶數的Vampire Number。

 例如：1260=21\*60或329346=342\*963，所以1260和329346 是Vampire Number。

**題意範例：**

4 🡺 1260

 1530

 6880

**解法：**

 計算兩個n/2位數相乘的每一種狀況，但是可以把不需要計算的直接跳過。

例如：因為題目只需要偶數，所以當遇到奇數乘奇數時可以直接跳過。相乘的兩數不能都是以0結尾，所以兩數都可以被10整除的話就可以直接跳過。A\*B 與B\*A是一樣的，所以不用重複算。

 設1000a+100b+10c+d 是Vampire Number

如果1000a +100b +10c +d=(10a +b) \*(10c +d)

1000a +100b +10c +d -(10a +b) -(10c +d)=990a +99b

可以被9整除

又如果1000a +100b +10c +d =(10b +a) \*(10d +c)

1000a +100b +10c +d -(10b +a) -(10d +c)=999a +90b +9c -9d

也可以被9整除。

多嘗試過幾種創況後可以發現，Vampire Number自身減掉構成自己的那兩個數字必定可以被9整除，所以在算迴圈時可以先將兩樹相乘再個別減掉那兩個數來判斷，如果結果是可以被9整除的那就可以繼續算下去。

**解法範例：**

340\*963=327420

 327420 -340 -963 =326117

 326117 = 9 \*36235 +2 不用算

341\*963 奇數，不用算

342\*963=329346

 329346 -342 -963 =328041

 328041 =9 \*36449

 可能是Vampire Number，可以繼續算算看

**討論：**

 無。

**程式：**

#include<iostream>

#include<cstdlib>

#include<cmath>

using namespace std;

void resetDigit(int arr[]){

 for(int i=0;i<10;i++) arr[i]=0;

}

void countDigit(int a,int arr[]){//計算每個數字出現幾次

 while(a){

 arr[a%10]++;

 a/=10;

 }

}

int check(int a[],int b[]){//看兩陣列是否相同

 int y=1;

 for(int i=0;i<10;i++){

 if(a[i]!=b[i]){

 y=0;

 break;

 }

 }

 return y;

}

void quickSort(int array[],int Left,int Right){

 if(Left>=Right) return;

 int mid=Left;

 int left=Left;

 int right=Right;

 while(left<right){

 while(array[right]>=array[mid]&&left<right) right--;

 while(array[left]<=array[mid]&&left<right) left++;

 if(left<right) swap(array[left],array[right]);

 }

 swap(array[mid],array[right]);

 quickSort(array,Left,right-1);

 quickSort(array,right+1,Right);

}

void printArray(int arr[],int length){

 int i,temp=0;

 for(i=0;i<length;i++){

 if(!arr[i]||arr[i]==temp) continue;

 temp=arr[i];

 cout<<arr[i]<<endl;

 }

 cout<<endl;

}

int main(){

 int digit[10]={0},digit2[10],input;

 int left,n,right,temp;

 int ans[3][2400]={0};

 int counter[3]={0};

 for(n=2;n<5;n++){

 counter[n-2]=0;

 for(right=pow(10,n-1);right<pow(10,n);right++){

 for(left=pow(10,n-1);left<=right;left++){

 if(left%2&&right%2||left%10==0&&right%10==0) continue;

 if((left\*right-left-right)%9!=0) continue;

 resetDigit(digit);

 resetDigit(digit2);

 temp=right\*left;

 countDigit(temp,digit);

 countDigit(left,digit2);

 countDigit(right,digit2);

 if(check(digit,digit2)){

 ans[n-2][counter[n-2]]=right\*left;

 counter[n-2]++;

 }

 }

 }

 quickSort(ans[n-2],0,counter[n-2]-1);

 }

 while(cin>>input){

 input=(input/2-2);

 printArray(ans[input],counter[input]);

 }

 return 0;

}