高精乘 FFT加速/Java大数

//HDU 1402 求高精度乘法

#include<cstdio>

#include<iostream>

#include<cmath>

#include<cstring>

#include<string>

using namespace std;

const double PI=acos(-1.0);

//复数结构体

struct Complex

{

double x,y;//实部和虚部 x+yi

Complex(double \_x = 0.0,double \_y = 0.0)

{

x = \_x;

y = \_y;

}

Complex operator - (const Complex &b)const

{

return Complex(x - b.x,y - b.y);

}

Complex operator +(const Complex &b)const

{

return Complex(x+b.x,y+b.y);

}

Complex operator \*(const Complex &b)const

{

return Complex(x\*b.x - y\*b.y,x\*b.y+y\*b.x);

}

};

/\*

\* 进行 FFT 和 IFFT 前的反转变换。

\* 位置 i 和（i 二进制反转后位置）互换

\* len 必须为 2 的幂

\*/

void change(Complex y[],int len)

{

int i,j,k;

for(i = 1,j = len/2; i<len-1; i++)

{

if(i < j)swap(y[i],y[j]);

//交换互为小标反转的元素，i<j 保证交换一次

//i 做正常的 +1，j 左反转类型的 +1, 始终保持 i 和 j 是反转的

k = len/2;

while(j >= k)

{

j-=k;

k/=2;

}

if(j < k)j += k;

}

}

/\*

\* 做 FFT

\* len 必须为2^k形式

\* on==1 时是 DFT，on==-1 时是 IDFT

\*/

void fft(Complex y[],int len,int on)

{

change(y,len);

for(int h = 2; h <= len; h <<= 1)

{

Complex wn(cos(-on\*2\*PI/h),sin(-on\*2\*PI/h));

for(int j = 0; j < len; j+=h)

{

Complex w(1,0);

for(int k = j; k < j+h/2; k++)

{

Complex u = y[k];

Complex t = w\*y[k+h/2];

y[k]=u+t;

y[k+h/2]=u-t;

w=w\*wn;

}

}

}

if(on==-1)

for(int i = 0; i < len; i++)

y[i].x /= len;

}

const int MAXN = 200010;

Complex x1[MAXN],x2[MAXN];

char str1[MAXN/2],str2[MAXN/2];

int sum[MAXN];

int main()

{

while(scanf("%s%s",str1,str2)==2)

{

int len1 = strlen(str1);

int len2 = strlen(str2);

int len = 1;

while(len < len1\*2 || len < len2\*2)

len<<=1;//len\*=2;

for(int i = 0; i < len1; i++)

x1[i] = Complex(str1[len1-1-i]-'0',0);//将数组反转

for(int i = len1; i < len; i++)

x1[i] = Complex(0,0);

for(int i = 0; i < len2; i++)

x2[i] = Complex(str2[len2-1-i]-'0',0);

for(int i = len2; i < len; i++)

x2[i] = Complex(0,0);

//求 DFT

fft(x1,len,1);

fft(x2,len,1);

for(int i = 0; i < len; ++i)

x1[i] = x1[i]\*x2[i];

fft(x1,len,-1);

for(int i = 0; i < len; ++i)

sum[i] = (int)(x1[i].x+0.5);

//到这里num里面保存的就是结果,就是乘积的系数向量

for(int i = 0; i < len; ++i)//处理进位

{

sum[i+1]+=sum[i]/10;

sum[i]%=10;

}

len = len1+len2-1;

while(sum[len] <= 0 && len > 0)//去前置0

--len;

for(int i = len; i >= 0; --i)

printf("%c",sum[i]+'0');

printf("\n");

}

return 0;

}