FFT 模板

A\*B problem 50000位

#include <stdio.h>

#include <string.h>

#include <iostream>

#include <algorithm>

#include <math.h>

using namespace std;

const double PI = acos(-1.0);

//复数结构体

struct complex

{

double r,i;

complex(double \_r = 0.0,double \_i = 0.0)

{

r = \_r; i = \_i;

}

complex operator +(const complex &b)

{

return complex(r+b.r,i+b.i);

}

complex operator -(const complex &b)

{

return complex(r-b.r,i-b.i);

}

complex operator \*(const complex &b)

{

return complex(r\*b.r-i\*b.i,r\*b.i+i\*b.r);

}

};

/\*

\* 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必须去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].r /= 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;

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

x1[i] = complex(str1[len1-1-i]-'0',0);

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].r+0.5);

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

{

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

sum[i]%=10;

}

len = len1+len2-1;

while(sum[len] <= 0 && len > 0)len--;

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

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

printf("\n");

}

return 0;

}