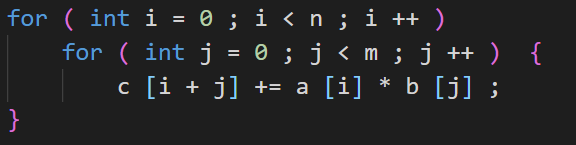
### 快速傅里叶变换——加速多项式乘法

班级：2018211302 学号：2018210913 姓名：杨成栋

1. **多项式乘法**

考虑到两个多项式的乘积，假设的项数为，其系数构成的维向量为，的项数为，其系数构成的维向量为。

则为求的系数构成的维的向量，考虑暴力求解，算法如下：



其时间复杂度是。

1. **多项式表示**

一个项数为的多项式可以用系数向量表示也可用表示。

证明过程略去。

1. **复数单位根的性质**

将复平面中的单位圆等分为部分，这个单位根为

其中幅角为正且最小的向量称为次单位向量，记为

对于单位根有两个性质：

1. 折半引理：



1. 消去引理：



1. **离散傅里叶变换（DFT）**

考虑一个项多项式，其中，其系数构成的维向量为。

我们将单位根的次幂分别带入得到其点值向量，这个过程成为离散傅里叶变换。

如果使用朴素算法求解，其时间复杂度为。

1. **快速傅里叶变换（FFT）**

利用单位根的性质，对于

考虑将其按照奇偶分组，即：



令





则有：



当时



当时



所以，如果我们已知分别在的取值，可以在的时间内求出的取值

则时间复杂度

1. **快速傅里叶逆变换（IFFT）**

使用快速傅里叶变换将点值表示的多项式转化为系数表示，这个过程叫做离散傅里叶反变换，即维点值向量推出维系数向量的过程，即在FFT的基础上再次进行FFT。

1. **快速傅里叶变换的C++实现**

利用快速傅里叶变换加速多项式相乘，思路就是先将读入的多项式系数表示转为点值表示，将多项式的点值表示相乘，最后利用傅里叶反变换将点值表示转化为系数表示。

以下是加速多项式相乘的代码：

#include<cstdio>

#include<complex>

using namespace std;

typedef complex<double> cp;

#define N 2097153

const double pie=acos(-1);

int n;

cp a[N],b[N];

int rev[N],ans[N];

char s1[N],s2[N];

//读入

int read(){

    int sum=0,f=1;

    char ch=getchar();

    while(ch>'9'||ch<'0'){if(ch=='-')f=-1;ch=getchar();}

    while(ch>='0'&&ch<='9'){sum=(sum<<3)+(sum<<1)+ch-'0';ch=getchar();}

    return sum\*f;

}

//初始化

void init(int k){

    int len=1<<k;

    for(int i=0;i<len;i++)

    rev[i]=(rev[i>>1]>>1)|((i&1)<<(k-1));

}

//FFT

void fft(cp \*a,int n,int flag){

    for(int i=0;i<n;i++)

    {

      if(i<rev[i])swap(a[i],a[rev[i]]);

    }

    for(int h=1;h<n;h\*=2)

    {

    cp wn=exp(cp(0,flag\*pie/h));

     for(int j=0;j<n;j+=h\*2)

     {

      cp w(1,0);

       for(int k=j;k<j+h;k++)

       {

         cp x=a[k];

         cp y=w\*a[k+h];

         a[k]=x+y;

         a[k+h]=x-y;

         w\*=wn;

       }

     }

     }

     if(flag==-1)

     for(int i=0;i<n;i++)

     a[i]/=n;

}

int main(){

    n=read();

    scanf("%s%s",s1,s2);

    for(int i=0;i<n;i++)a[i]=(double)(s1[n-i-1]-'0');

    for(int i=0;i<n;i++)b[i]=(double)(s2[n-i-1]-'0');

    int k=1,s=2;

    while((1<<k)<2\*n-1)k++,s<<=1;

init(k);

//把a的系数表示转化为点值表示

fft(a,s,1);

//把b的系数表示转化为点值表示

fft(b,s,1);

//两个多项式的点值表示相乘

    for(int i=0;i<s;i++)

a[i]\*=b[i];

//把这个点值表示转化为系数表示

    fft(a,s,-1);

    for(int i=0;i<s;i++)

    {

    ans[i]+=(int)(a[i].real()+0.5);

    ans[i+1]+=ans[i]/10;

    ans[i]%=10;

    }

    while(!ans[s]&&s>-1)s--;

    if(s==-1)printf("0");

    else

    for(int i=s;i>=0;i--)

    printf("%d",ans[i]);

    return 0;

}