

# 快速傅里叶变换(FFT)

## 原理

[OI Wiki-快速傅里叶变换](#)

## 例题

### 例 1

#### 题意

[P3803 【模板】多项式乘法 FFT](#)

给定一个  $n$  次多项式  $F(x)$  和一个  $m$  次多项式  $G(x)$  求出  $F(x)$  和  $G(x)$  的卷积。

#### 题解

通过  $DFT$  和  $IDFT$  两个过程，实现多项式由系数表示法 点值表示法 系数表示法。利用单位根的性质实现奇偶分治过程，时间复杂度  $O(n \log n)$  其中递归的分治过程时空消耗较大，通过蝴蝶变换找到各项系数（点值）分治后的位置，直接往上迭代，实现优化。

#### 评价

FFT 模板题

#### 代码

```
#include<bits/stdc++.h>
using namespace std;
typedef double db;
typedef complex<db> Comp;
const db PI=acos(-1.0);
const int N=1e7+5;
int rev[N];// i 二进制翻转后的数
Comp a[N],b[N];// 系数(点值)
void FFT(Comp *y,int len,int on){
    for(int i=0;i<len;i++)
        if(i<rev[i])
            swap(y[i],y[rev[i]]);// 变到分治后的位置,准备往上迭代
    for(int mid=1;mid<len;mid<<=1){// mid 是区间长度的一半
        Comp wn(cos(PI/mid),sin(on*PI/mid));// 单位根
```



项式相乘的问题，套用 FFT 模板即可，时间复杂度  $O(n \log n)$  ( $n$  为位数)。

## 评价

## FFT 模板题

## 代码

```
#include<bits/stdc++.h>
using namespace std;
typedef double db;
typedef complex<db> Comp;
const db PI=acos(-1.0);
const int N=1e7+5;
int rev[N];
Comp a[N],b[N];
int c[N];
char s1[N],s2[N];
void FFT(Comp *y,int len,int on){// FFT 模板
    for(int i=0;i<len;i++)
        if(i<rev[i])
            swap(y[i],y[rev[i]]);
    for(int mid=1;mid<len;mid<=<=1){
        Comp wn(cos(PI/mid),sin(on*PI/mid));
        for(int j=0,R=mid<<1;j<len;j+=R){
            Comp w(1,0);
            for(int k=0;k<mid;k++,w=w*wn){
                Comp u=y[j+k];
                Comp t=w*y[j+k+mid];
                y[j+k]=u+t;
                y[j+k+mid]=u-t;
            }
        }
    }
}
int main(){
    scanf("%s%s",s1,s2);
    int n=strlen(s1),m=strlen(s2);
    n--,m--;
    for(int i=0;i<=n;i++) a[i]={(double)(s1[n-i]-'0'),0};
    for(int i=0;i<=m;i++) b[i]={(double)(s2[m-i]-'0'),0};
    int len=1,l=0;
    while(len<=n+m) len<=<=1,l++;
    for(int i=0;i<len;i++) rev[i]=(rev[i>>1]>>1)|((i&1)<<(l-1));
    FFT(a,len,1);
    FFT(b,len,1);
    for(int i=0;i<=len;i++) a[i]=a[i]*b[i];
    FFT(a,len,-1);
}
```

```
for(int i=0;i<=n+m;i++){// 需要进位
    int x=(int)(real(a[i])/len+0.5);
    c[i]+=x;
    c[i+1]+=c[i]/10;
    c[i]=c[i]%10;
}
if(c[n+m+1]!=0) n++;
for(int i=n+m;i>=0;i--)
    printf("%d",c[i]);
return 0;
}
```