数学

1. 快速幂

```
int qpow(int a,int n)
{
    int ans=1;
    while(n)
    {
        if(n&1)
        {
            ans =ans*a%mod;
        }
        a=a*a%mod;
        n>>=1;

    }
    return ans;
}
```

2. exgcd

```
int exgcd(int a,int b,int &x,int &y)
{
    if(b==0)
    {
        x=1;
        y=0;
        return a;
    }
    int d=exgcd(b,a%b,x,y),x0=x,y0=y;
    x=y0;
    y=x0-(a/b)*y0;
    return d;
}
```

3.线性inv

```
void getinv(int n)
{
    inv[1]=1;
    for(int i=2;i<=n;i++)
    {
        inv[i]=mod-((mod/i)*inv[mod%i])%mod;
    }
}
```

4.数论分块

```
int ans=0;
    for(int l=1,r;l<=n;l=r+1)
    {
        r=n/(n/l);
        ans+=(r-l+1)*(n/l);
    }
    cout<<ans<<endl;
```

5.欧拉筛

```
int Eular(int n) {
    int cnt = 0;
    memset(is_prime, true, sizeof(is_prime));
    is_prime[0] = is_prime[1] = false;
    for(int i=2;i<=n;i++)
    {
        if(is_prime[i])
        {
            prime[++cnt]=i;
        }
        for(int j=1;j<=cnt&&i*prime[j]<=n;j++)
        {
            is_prime[i*prime[j]]=0;
            if(i%prime[j]==0)break;
        }
    }
    return cnt;
}
```

6.欧拉函数

```
int Eular(int n) {
    int cnt = 0;
    memset(is_prime, true, sizeof(is_prime));
    is_prime[0] = is_prime[1] = false;
    for(int i=2;i<=n;i++)
    {
        if(is_prime[i])
        {
            prime[++cnt]=i;
        }
        for(int j=1;j<=cnt&&i*prime[j]<=n;j++)
        {
            is_prime[i*prime[j]]=0;
            if(i%prime[j]==0)break;
        }
    }
    return cnt;
}
```

7.组合数

```
int fac[N];
int  inv[N];
void init(int n)
{
    fac[0] = 1;
    inv[0] = 1;
    inv[1] = 1;
    fac[1] = 1;
    for(int i = 2;i<=2*n;i++)
    {
        fac[i] = fac[i-1]*i%mod;
        inv[i] = (mod-mod/i)*inv[mod%i]%mod;
    }
    for(int i = 1;i<=n;i++)
    {
        inv[i] = inv[i]*inv[i-1]%mod;
    }
}
int C(int n,int m)
{
    if(m>n||m<0||n<0)return 0;
    return fac[n]*inv[m]%mod*inv[n-m]%mod;
}
```

8.欧拉定理

a在mod m意义下，

$a^{(b}c)$ 与 a^(b mod(eular(m))+m)同余

9.卡特兰数

```
int C(int n,int m)
{
    return fac[n]*qpow(fac[n-m],mod-2)%mod*qpow(fac[m],mod-2)%mod;
}
int cat(int n)
{
    return C(2*n,n)*qpow(n+1,mod-2)%mod;
}
```

10. 矩阵

```
    class matrix
{
    public:
        int x[105][105];
        int sz;
        matrix(int n)
        {sz=n;
```

```
            for(int i=1;i<=sz;i++)
            {
                for(int j=1;j<=sz;j++)
                {
                    x[i][j]=0;
                }
            }
        }
        matrix mul(matrix a,matrix b);
        matrix qpow(matrix a,int n);
        void tra(matrix a);
};

matrix matrix::mul(matrix a, matrix b) {
    matrix c(a.sz);
    for(int i=1;i<=a.sz;i++)
        for(int j=1;j<=a.sz;j++)
            for(int k=1;k<=a.sz;k++)
                c.x[i][j]=(c.x[i][j]%mod+(a.x[i][k]*b.x[k][j])%mod)%mod;
    return c;
}
matrix matrix::qpow(matrix a,int n)
{
    matrix res(a.sz);
    for(int i=1;i<=a.sz;i++)res.x[i][i]=1;
    while(n>0)
    {
        if(n&1)res= mul(res,a);
        a= mul(a,a);
        n>>=1;
    }
    return res;
}
void matrix::tra(matrix a) {
    for(int i=1;i<=a.sz;i++)
    {
        for(int j=1;j<=a.sz;j++)
        {
            cout<<a.x[i][j]<<" ";
        }
        cout<<endl;
    }
}
```