拉格朗日插值

有n个点$(x_1,y_1),(x_2,y_2)\cdots(x_n,y_n)$，满足$\forall i\neq j, x_i\neq x_j$。你需要求出一个次数小于等于$n-1$的多项式$f(x)$，使得$\forall i\in[1,n], f(x_i)=y_i$。

基本原理

考虑对于每一个i，构造出一个在x_i处取值为1、在另外$n-1$个点取值为0的多项式$\ell_i(x)$：

$$\ell_i(x)=\frac{\prod_{j\neq i}(x-x_j)}{\prod_{j\neq i}(x_i-x_j)}$$

这个多项式显然是满足上面的条件的。

由此我们可以构造出一个多项式$L(x)=\sum_{i=1}^n y_i\ell_i(x)$。这个多项式显然满足在$x_i$点取值为$y_i$。

唯一性证明

假设存在两个次数小于等于$n-1$的满足条件的多项式P_1和P_2，那么P_1-P_2在$x_1,x_2\cdots x_n$的取值一定都是0，也就是说这个多项式是$(x-x_1)(x-x_2)\cdots(x-x_n)$的倍数。考虑到$P_1-P_2$的次数不可能超过$n-1$，而$(x-x_1)(x-x_2)\cdots(x-x_n)$是一个$n$次多项式，所以$P_1-P_2$一定是这个多项式的0倍，也就是说$P_1-P_2=0$，也就是$P_1=P_2$。

实现

一般我们是需要求出将某个值带入这个多项式得到的值。这个时候我们就没有必要求多项式的表达式了，直接把值带入式子里面算，时间复杂度$O(n^2)$。

```
int main() {
    rd(n),rd(k);
    for(int i=1;i<=n;++i) rd(x[i]),rd(y[i]);
    int Ans=0;
    for(int i=1;i<=n;++i) {
        int d=1,f=1;
        for(int j=1;j<=n;++j)
            if(i!=j) {
                d=d*(ll)(k-x[j])%mod;
                f=f*(ll)(x[i]-x[j])%mod;
            }
        f=Pow(f,mod-2);
        Ans=(Ans+y[i]*(ll)f%mod*d)%mod;
    }
    printf("%lld\n",(Ans+mod)%mod);
    return 0;
}
```

取值连续时的优化

当x的取值是连续的一段自然数的时候，我们可以算出$(x-x_i)$的前缀积和后缀积，就可以在$O(1)$的时间内得到分子。观察发现ℓ_i分母的绝对值是$\frac{1}{(i-1)!(n-i)!}$，它的符号与$n-i$的奇偶性相关。预处理出阶乘的逆元也就可以$O(1)$算了。