A. Orange与算术问题

```
int factor[100000];
void divide(int u,int op)
{
    for(int i = 2; i <= u / i; i ++)
    {
        if(u % i == 0)
        {
            int cnt = 0;
            while(u % i == 0) cnt++, u /= i;
            factor[i] += cnt * op;
        }
    }
    if(u > 1)
    {
        factor[u] += op;
    }
}

signed main()
{
    IOS;
    int n;
    cin >> n;
    while(n--)
    {
        int op, x;
        cin >> op >> x;
        divide(x, op == 1 ? 1 : -1);
    }
    cin >> n;
    while(n--)
    {
        int op, x;
        cin >> op >> x;
        divide(x, op == 2 ? 1 : -1);
    }
    if(*max_element(factor, factor + 100000) == 0) cout << "YES\n";
    else cout << "NO\n";
    return 0;
}
```

B. 四两拨千斤

```
int vis[50001], vis2[1000001];
vector<int> primes;
signed main()
{
    IOS;
    int l, r;
    cin >> l >> r;
    for(int i = 2; i <= 50000; i ++)
    {
        if(!vis[i]) primes.push_back(i);
        for(int p : primes)
        {
            if(p * i > 50000) break;
            vis[p * i] = 1;
            if(i % p == 0) break;
        }
    }
    for(auto p : primes)
    {
        for(int i = max((l + p - 1) / p * p, 2 * p); i <= r / p * p; i += p)
            vis2[i - l] = 1;
    }
    int ans = 0;
    for(int i = 0; i <= (r - l); i ++)
        ans += vis2[i] == 0;
    cout << ans << '\n';
    return 0;
}
```

C. 约数之和

```
map<int,int> factor;
void divide(int u)
{
    for(int i = 2; i <= u / i; i ++)
    {
        if(u % i == 0)
        {
            int cnt = 0;
            while(u % i == 0) cnt++, u /= i;
            factor[i] += cnt;
        }
    }
    if(u > 1)
    {
        factor[u] += 1;
    }
}

signed main()
{
    IOS;
    int n;
    cin >> n;
    while(n--)
    {
        int x;
        cin >> x;
        divide(x);
    }
    i64 ans = 1;
    for(auto[p, c] : factor)
    {
        i64 temp = 1;
        while(c--) temp = (temp * p + 1) % mod;
        ans = ans * temp % mod;
    }
    cout << ans << '\n';
    return 0;
}
```

D. 质数距离

```
vector<int> primes;
int vis[N];

void getpirme(int n)
{
    for (int i = 2; i <= n; i++)
    {
        if (!vis[i])
            primes.push_back(i);
        for (int j = 0; primes[j] * i <= n; j++)
        {
            vis[primes[j] * i] = 1;
            if (i % primes[j] == 0)
                break;
        }
    }
}

signed main()
{
    int limit = 50000;
    getpirme(50000);
    i64 l, r;
    while (cin >> l >> r)
    {
        vector<int> vvis(r - l + 2), mprimes;
        for (int p : primes)
        {
            for (i64 i = max((l + p - 1) / p * p, 2ll * p); i <= r; i += p)
            {
                vvis[i - l] = 1;
            }
        }
        for (i64 i = l; i <= r; i++)
        {
            if ((!vvis[i - l] || (i < limit && !vis[i])) && i >= 2)
            {
                mprimes.push_back(i);
            }
        }
        if (mprimes.size() < 2)
        {
            cout << "There are no adjacent primes." << endl;
            continue;
```

```
        }
        int mx = 0, mi = 0;
        for (int i = 0; i < mprimes.size() - 1; i++)
        {
            if (mprimes[mx + 1] - mprimes[mx] < mprimes[i + 1] - mprimes[i])
            {
                mx = i;
            }
            if (mprimes[mi + 1] - mprimes[mi] > mprimes[i + 1] - mprimes[i])
            {
                mi = i;
            }
        }
        cout << mprimes[mi] << ',' << mprimes[mi + 1] << " are closest, " << mprimes[mx] << ',
    }
    return 0;
}
```

E. Orange的神奇盒子

```
int vis[N], st[N];

signed main()
{
    IOS;
    int n;
    cin >> n;
    vector<int> a(n + 1);
    int flag = 0;
    for(int i = 1; i <= n; i ++)
    {
        cin >> a[i];
        flag += a[i] == 1;
        st[a[i]] = 1;
    }
    if(!flag)
    {
        cout << 0;
        return 0;
    }
    for(int i = 2; i <= N; i ++)
    {
        if(!st[i])
        {
            for(int j = i; j <= N; j += i)
            {
                vis[j] = 1;
            }
        }
    }
    int ans = 0;
    for(int i = 1; i <= N ; i ++)
    {
        if(!vis[i])
        {
            ans ++;
        }
    }
    cout << ans << '\n';
    return 0;
}
```

F. 数星星

```
int fact[N], inv[N], d[N];

int qmi(int a, int k, int p = mod)
{
    int res = 1;
    while (k)
    {
        if (k & 1)
            res = (i64)res * a % p;
        k >>= 1;
        a = (i64)a * a % p;
    }
    return res;
}

int C(int n, int m)
{
    if (m > n)
        return 0;
    if (n == m)
        return 1;
    return (i64)fact[n] * inv[m] % mod * inv[n - m] % mod;
}

signed main()
{
    fact[0] = 1;
    inv[0] = 1;
    for (int i = 1; i < N; i++)
    {
        fact[i] = (i64)fact[i - 1] * i % mod;
        inv[i] = qmi(fact[i], mod - 2);
    }
    int n;
    cin >> n;
    for (int i = 1; i < n; i++)
    {
        int u, v;
        cin >> u >> v;
        d[u]++, d[v]++;
    }
    int L, R;
    i64 ans = 0;
    cin >> L >> R;
```

```
    if (L == 1)
    {
        ans += n;
        L++;
    }
    if (L == 2)
    {
        ans += n - 1;
        L++;
    }
    for (int i = 1; i <= n; i++)
    {
        for (int k = L - 1; k <= R - 1 && k <= d[i]; k++)
        {
            ans = (ans + C(d[i], k)) % mod;
        }
    }
    cout << ans;
    return 0;
}
```