1. Approximate K-th Root using powl (Less Accurate – Fast)

```
// Returns floor of the k-th root of n (i.e., max x such that x^k ≤ n)
// May suffer from floating point precision errors for large numbers
ll kthRoot_approx(ll n, ll k) {
    double root = powl((double)n, 1.0 / (double)k);
    ll realRoot = (ll)(root - 1);

    while (true) {
        ll a = realRoot + 1, p = 1;
        // Compute a^k while checking for overflow
        for (int j = 0; j < k; j++) {
            if (p > n / a) // overflow or exceeds n
                return realRoot;
            p *= a;
        }
        ++realRoot;
    }
}
```

---

2. Accurate K-th Root using Binary Search (Recommended)

```
// Accurate method using binary search to compute floor(k-th root of n)
ll kthRoot(ll n, ll k) {
    ll low = 1, high = n, ans = 1;

    while (low <= high) {
        ll mid = low + (high - low) / 2;
        __int128 p = 1;
        bool overflow = false;

        // Compute mid^k safely using __int128 to avoid overflow
        for (int i = 0; i < k; i++) {
            p *= mid;
            if (p > n) {
                overflow = true;
                break;
            }
        }

        if (!overflow) {
            ans = mid;       // mid^k <= n → valid candidate
            low = mid + 1;   // try larger
        } else {
            high = mid - 1;  // mid^k > n → too big
        }
    }

    return ans;
}
```

استخدم `kthRoot(n, k)` لو عايزة نتيجة دقيقة وآمنة من overflow.
استخدم `kthRoot_approx(n, k)` لو السرعة أهم والـ `n` صغيرة أو مش بتفرق.