FactoringLargeNumbers.cpp

```cpp
#include<bits/stdc++.h>
#include<time.h>

#define show(x) cout << #x << " = " << x << endl;

using namespace std;

typedef long long ll;
typedef pair<ll, ll> ii;
typedef pair<double, ii> iii;

const int MAX = 200005;
const double EPS = 1e-5;
const int INF = INT_MAX;

ll mul(ll a, ll b, ll mod) {
    ll ret = 0;
    for(a %= mod, b %= mod; b != 0;
        b >>= 1, a <<= 1, a = a >= mod ? a - mod : a) {
        if (b&1) {
            ret += a;
            if (ret >= mod) ret -= mod;
        }
    }
    return ret;
}

ll fpow(ll a, ll b, ll MOD) {
    ll ans = 1LL;
    while(b > 0) {
        if(b&1) ans = mul(ans, a, MOD);
        a = mul(a, a, MOD);
        b >>= 1LL;
    }
    return ans;
}

const int rounds = 6;
// Checks if a number is prime with prob 1 - 1 / (2 ^ it)
bool miller_rabin(ll n) {
    if(n == 2 || n == 3) return true;
    if(n < 2 || (n&1) == 0) return false;
    for(int i = 0; i < rounds; i++) {
        int a = rand()%(n-4)+2;
        if(fpow(a, n-1, n) != 1)
            return false;
    }
    return true;
}

// if n is prime , check with miller rabin (n^(1/4)) and check return != n and  != 1
ll pollard_rho(ll n, ll c) {
    ll x = 2, y = 2, i = 1, k = 2, d;
    while (true) {
        x = (mul(x, x, n) + c);
        if (x >= n) x -= n;
        d = __gcd(x - y, n);
        if (d > 1) return d;
        if (++i == k) y = x, k <<= 1;
    }
    return n;
}

void factorize(ll n, vector<ll> &f) {
    if(n == 1) return;
    if (miller_rabin(n)) {
        f.push_back(n);
        return;
    }
    ll d = n;
    for (int i = 2; d == n; i++)
        d = pollard_rho(n, i);
    factorize(d, f);
    factorize(n/d, f);
}

int main() {
    ll n;
    while(cin >> n && n != -1) {
        vector<ll> ans;
        factorize(n, ans);
        sort(ans.begin(), ans.end());
        for(int i = 0; i < ans.size(); i++)
            cout << "    " << ans[i] << "\n";
        cout << "\n";
    }
}
```