C1J 魔法糖果

问题分析

从 `x = 1` 开始，每次可以选择将当前数变为 `2 * x + 1` 或者 `2 * x - 1`

对于一个输入 `t`，我们是否可以通过最少此操作使得 `x` 从 `1` 变为 `t`

题解思路

观察一

观察 `x` 的变化方式，我们可以轻松观察到，`x` 在变化过程中一直是**奇数**

如：

$$1 \to^{\times 2+1} 3 \to^{\times 2-1} 5 \to^{\times 2+1} 11 \cdots$$

故我们可以**断定**，**对于所有的 `t % 2 == 0` ，一定无法通过有限次操作得到！**

观察二

手玩几个样例：

$$1, 3, 5, 7, 9, 11, \cdots$$

都可以实现，我们**猜测**：**所有的奇数一定都可以通过有限次操作得到！**

观察三

我们考虑从已知入手，从给定值逆向尝试如何回到 **1**

对于一个给定的奇数，我们考虑如何找到它的上一步

因为过程中一定是所有数都是**奇数**，且通过两种操作 `2 * x + 1` 与 `2 * x - 1` 得到，对于任意一个奇数，我们一定可以设 `t = 4 * k + p` 其中 `p = 1 or p = 3`

- 若 `t = 4 * k + 1` ，
    - 那么 `(t - 1) / 2 = 2 * k` 一定是一个偶数，**故此时的 `t` 一定不是通过这一步得到！**
    - 但 `(t + 1) / 2 = 2 * k + 1` ，一定是一个奇数，**此时的 `t` 一定是通过这一步得到！**
- 若 `t = 4 * k + 3` ，

- 那么 `(t - 1) / 2 = 2 * k + 1` 一定是一个奇数，此时的 `t` 一定是通过这一步得到！
- 但 `(t + 1) / 2 = 2 * k + 2` 一定是一个偶数，此时的 `t` 一定不是通过这一步得到！

综上所述：对于任意一步的 `t`，一定只有一种途径可以转移到它

故其实不存在最短方法了，得到的路径只有一条

数据结构

无特殊数据结构，只需要 `while` 循环或者递归即可

伪代码

```
function solve (n):
    insert n into ansArray
    if n = 1 return
    if (n - 1) / 2 is odd:
        solve((n - 1) / 2)
    else:
        solve((n + 1) / 2)
```

算法分析

由于每次进行操作，`t` 都会减少一半，故时间复杂度：$O(\log n)$

代码实现

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

using namespace std;

using ll = long long;
using db = long double;  // or double, if TL is tight
using str = string;
using u32 = unsigned int;
using u64 = unsigned long long;
using u128 = unsigned __int128;

using pii = pair<int, int>;
using pll = pair<ll, ll>;
using pdd = pair<db, db>;
#define fi first
#define se second

#define vc vector
using vi = vc<int>;
using vb = vc<bool>;
using vll = vc<ll>;
```

```
using vd = vc<db>;
using vs = vc<str>;
using vch = vc<char>;
using vpii = vc<pii>;
using vpll = vc<pll>;
using vpdd = vc<pdd>;
using vvi = vc<vc<int>>;
using vvll = vc<vc<ll>>;
using vvch = vc<vc<char>>;
using vvb = vc<vc<bool>>;
using vvpii = vc<vc<pii>>;
using vvpll = vc<vc<pll>>;

const int mod = 998244353;  // 1e9 + 7;
const int INF = 0x3f3f3f3f;
const ll BIG = 1e18;  // not too close to LLONG_MAX
const db PI = acos((db)-1);
const int dx[4]{1, 0, -1, 0}, dy[4]{0, 1, 0, -1};  // for
every grid problem!!
mt19937
rng((uint32_t)chrono::steady_clock::now().time_since_epoch
().count());

#define cl(x, y) memset(x, y, sizeof(x))
#define de(x) cerr << #x << " = " << x << " "
#define del(x) cerr << #x << " = " << x << endl

#define overload3(a, b, c, d, ...) d
#define rep1(a) for(int i = 0; i < (a); i++)
#define rep2(i, a) for (int i = 0; i < (a); i++)
#define rep3(i, a, b) for (int i = (a); i <= (b); i++)
#define per1(a) for(int i = (a) - 1; i >= 0; i--)
#define per2(i, a) for (int i = (a) - 1; i >= 0; i--)
#define per3(i, a, b) for (int i = (a); i >= (b); i--)
#define rep(args...) overload3(args, rep3, rep2, rep1)
(args)
#define per(args...) overload3(args, per3, per2, per1)
(args)
#define for_subset(t, s) \
    for (ll t = (s); t >= 0; t = (t == 0 ? -1 : (t - 1) &
(s)))

#define co cout
#define dl endl
#define ret return
#define pb push_back
#define YES cout << "YES" << endl
#define Yes cout << "Yes" << endl
#define NO cout << "NO" << endl
#define No cout << "No" << endl

#define all(s) s.begin(), s.end()
```

```
#define rall(s) s.rbegin(), s.rend()
#define sz(v) int(v.size())
#define SUM(a) accumulate(all(a), 0ll)
#define MIN(v) (*min_element(all(v)))
#define MAX(v) (*max_element(all(v)))
#define unisort(v) \
    sort(all(v));  \
    v.resize(unique(all(v)) - v.begin())

int popcnt(int x) { return __builtin_popcount(x); }
int popcnt(u32 x) { return __builtin_popcount(x); }
int popcnt(ll x) { return __builtin_popcountll(x); }
int popcnt(u64 x) { return __builtin_popcountll(x); }

#define LOWBIT(x) ((x) & (-x))
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 1, 2)
int topbit(int x) { return (x == 0 ? -1 : 31 -
__builtin_clz(x)); }
int topbit(u32 x) { return (x == 0 ? -1 : 31 -
__builtin_clz(x)); }
int topbit(ll x) { return (x == 0 ? -1 : 63 -
__builtin_clzll(x)); }
int topbit(u64 x) { return (x == 0 ? -1 : 63 -
__builtin_clzll(x)); }
// (0, 1, 2, 3, 4) -> (-1, 0, 1, 0, 2)
int lowbit(int x) { return (x == 0 ? -1 :
__builtin_ctz(x)); }
int lowbit(u32 x) { return (x == 0 ? -1 :
__builtin_ctz(x)); }
int lowbit(ll x) { return (x == 0 ? -1 :
__builtin_ctzll(x)); }
int lowbit(u64 x) { return (x == 0 ? -1 :
__builtin_ctzll(x)); }
template <typename T> T cdiv(T &a, T &b) {
    return a / b + ((a ^ b) > (T)(0) && a % b);
}  // divide a by b rounded up
template <typename T> T fdiv(T &a, T &b) {
    return a / b - ((a ^ b) < (T)(0) && a % b);
}  // divide a by b rounded down

#define LOCAL_TEST freopen("out.txt", "w", stdout)
#define IOS ios::sync_with_stdio(false);cin.tie(0)

const int maxn = 1e5 + 5;

inline void INIT() {

}

vi ans;
inline void DO(int n) {
    ans.pb(n);
```

```
    if(n == 1) ret;
    if(((n - 1) / 2) % 2) {
        DO((n - 1) / 2);
    } else {
        DO((n + 1) / 2);
    }
}

inline void SOLVE() {

    int n; cin >> n;
    if(n % 2 == 0) { co << -1 << dl; ret; }
    DO(n);

    reverse(all(ans));
    for(auto i: ans) co << i << " ";
    co << dl;

}

int main() {

    IOS;

    INIT();
    int tt = 1;
    // cin >> tt;
    while(tt--)
        SOLVE();

   //  system("pause");
    return 0;
}
```