Problem
Odin has some magical rings which produce copies of themselves. Each "X-day ring" produces one more X-day ring every X days after the day it came into existence. These rings come in six possible varieties: 1-day, 2-day, ..., all the
ay up to 6-day.

For example, a 3-day ring that came into existence on day 0 would do nothing until day 3, when it would produce an
ther 3-day ring. Then, on day 6, each of those two rings would produce another 3-day ring, and so on.

You know that Odin had no rings before day 0. On day 0, some rings came into existence. At the end of day 0, Odin
ad Ri i-day rings, for each $1 \leq i \leq 6$. You know that $0 \leq Ri \leq 100$, for all i, and at least one of the Ri values is positiv
.

Fortunately, you also have access to the secret well of knowledge. Each time you use it, you can find out the total nu
ber of rings that Odin had at the end of a particular day between day 1 and day 500, inclusive. The well will give yo
the answer modulo 263, because even it can only hold so much information! Moreover, you can only use the well u
to W times.

Your goal is to determine how many rings of each type Odin had at the end of day 0 — that is, you must find each o
the Ri values.

Input and output
This is an interactive problem. You should make sure you have read the information in the Interactive Problems sect
on of our FAQ.

Initially, your program should read a single line containing two integers T, the number of test cases, and W, the num
er of times you are allowed to use the well of knowledge per test case. Then, you need to process T test cases.

In each test case, your program processes up to W + 1 exchanges with our judge. You may make up to W exchanges of the following form:

Your program outputs one line with a single integer D between 1 and 500, inclusive.
The judge responds with one line with a single integer: the total number of rings that Odin had at the end of day D,
odulo 263. If you send invalid data (e.g., a number out of range, or a malformed line), the judge instead responds wit
-1.
After between 0 and W exchanges as explained above, you must perform one more exchange of the following form:

Your program outputs one line with six integers R1, R2, R3, R4, R5, R6, where Ri represents the number of i-day ri
gs that Odin had at the end of day 0.
The judge responds with one line with a single integer: 1 if your answer is correct, and -1 if it is not (or you have pro
ided a malformed line).
After the judge sends -1 to your input stream (because of either invalid data or an incorrect answer), it will not send
ny other output. If your program continues to wait for the judge after receiving -1, your program will time out, result
ng in a Time Limit Exceeded error. Notice that it is your responsibility to have your program exit in time to receive
Wrong Answer judgment instead of a Time Limit Exceeded error. As usual, if the memory limit is exceeded, or you
program gets a runtime error, you will receive the appropriate judgment.

Limits
$1 \leq T \leq 50$.
Time limit: 20 seconds per test set.
Memory limit: 1GB.

Test set 1 (Visible)

W = 6.

Test set 2 (Hidden)
W = 2.

Testing Tool
You can use this testing tool to test locally or on our servers. To test locally, you will need to run the tool in parallel ith your code; you can use our interactive runner for that. For more information, read the Interactive Problems sectio of the FAQ.

Local Testing Tool
To better facilitate local testing, we provide you the following script. Instructions are included inside. You are encou aged to add more test cases for better testing. Please be advised that although the testing tool is intended to simulate he judging system, it is NOT the real judging system and might behave differently.

If your code passes the testing tool but fails the real judge, please check the Coding section of our FAQ to make sure that you are using the same compiler as us.

Sample Interaction
This interaction corresponds to Test set 1. Suppose that, unbeknownst to us, the judge has decided that Odin had one ring of each of the six types at the end of day 0.

```
  t, w = readline_int_list()   // Reads 50 into t and 6 into w
  printline 3 to stdout        // Asks about day 3.
  flush stdout
  n = readline_int()           // Reads 15 into n.
  printline 1 to stdout        // Asks about day 1.
  flush stdout
  n = readline_int()           // Reads 7 into n.
  printline 1 1 1 3 0 0 to stdout
  flush stdout                 // We make a guess even though we could have
                         // queried the well up to four more times.
  verdict = readline_int()     // Reads -1 into verdict (judge has decided our
                         //   solution is incorrect)
  exit                         // Exits to avoid an ambiguous TLE error
```
Notice that even though the guess was consistent with the information we received from the judge, we were still wro g because we did not find the correct values.

Solution:

```
#pragma GCC optimize ("O3")
#pragma GCC target ("sse4")

#include <bits/stdc++.h>
#include <ext/pb_ds/tree_policy.hpp>
#include <ext/pb_ds/assoc_container.hpp>
#include <ext/rope>

using namespace std;
using namespace __gnu_pbds;
using namespace __gnu_cxx;

typedef long long ll;
typedef long double ld;
typedef complex<ld> cd;

typedef pair<int, int> pi;
typedef pair<ll,ll> pl;
typedef pair<ld,ld> pd;

typedef vector<int> vi;
typedef vector<ld> vd;
typedef vector<ll> vl;
typedef vector<pi> vpi;
typedef vector<pl> vpl;
typedef vector<cd> vcd;

template <class T> using Tree = tree<T, null_type, less<T>, rb_tree_tag,tree_order_statistics_node_update>;

#define FOR(i, a, b) for (int i = (a); i < (b); i++)
#define F0R(i, a) for (int i = 0; i < (a); i++)
#define FORd(i,a,b) for (int i = (b)-1; i >= (a); i--)
#define F0Rd(i,a) for (int i = (a)-1; i >= 0; i--)
#define trav(a, x) for (auto& a : x)

#define mp make_pair
#define pb push_back
#define f first
#define s second
#define lb lower_bound
#define ub upper_bound

#define sz(x) (int)x.size()
#define beg(x) x.begin()
#define en(x) x.end()
#define all(x) beg(x), en(x)
#define resz resize

const int MOD = 1000000007; // 998244353
const ll INF = 1e18;
const int MX = 100005;
const ld PI = 4*atan((ld)1);

template<class T> void ckmin(T &a, T b) { a = min(a, b); }
template<class T> void ckmax(T &a, T b) { a = max(a, b); }

namespace input {
    template<class T> void re(complex<T>& x);
    template<class T1, class T2> void re(pair<T1,T2>& p);
    template<class T> void re(vector<T>& a);
    template<class T, size_t SZ> void re(array<T,SZ>& a);
    template<class T> void re(T& x) { cin >> x; }
    void re(double& x) { string t; re(t); x = stod(t); }
    void re(ld& x) { string t; re(t); x = stold(t); }
    template<class Arg, class... Args> void re(Arg& first, Args&... rest) {
        re(first); re(rest...);
    }

    template<class T> void re(complex<T>& x) { T a,b; re(a,b); x = cd(a,b); }
    template<class T1, class T2> void re(pair<T1,T2>& p) { re(p.f,p.s); }
    template<class T> void re(vector<T>& a) { F0R(i,sz(a)) re(a[i]); }
    template<class T, size_t SZ> void re(array<T,SZ>& a) { F0R(i,SZ) re(a[i]); }
}

using namespace input;

namespace output {
    template<class T1, class T2> void pr(const pair<T1,T2>& x);
    template<class T, size_t SZ> void pr(const array<T,SZ>& x);
    template<class T> void pr(const vector<T>& x);
    template<class T> void pr(const set<T>& x);
    template<class T1, class T2> void pr(const map<T1,T2>& x);

    template<class T> void pr(const T& x) { cout << x; }
    template<class Arg, class... Args> void pr(const Arg& first, const Args&... rest) {
        pr(first); pr(rest...);
    }

    template<class T1, class T2> void pr(const pair<T1,T2>& x) {
        pr("{",x.f,", ",x.s,"}");
    }
    template<class T> void prContain(const T& x) {
        pr("{");
        bool fst = 1; trav(a,x) pr(!fst?", ":"",a), fst = 0;
        pr("}");
    }
    template<class T, size_t SZ> void pr(const array<T,SZ>& x) { prContain(x); }
    template<class T> void pr(const vector<T>& x) { prContain(x); }
    template<class T> void pr(const set<T>& x) { prContain(x); }
    template<class T1, class T2> void pr(const map<T1,T2>& x) { prContain(x); }

    void ps() { pr("\n"); }
    template<class Arg, class... Args> void ps(const Arg& first, const Args&... rest) {
        pr(first," "); ps(rest...); // print w/ spaces
    }
}

using namespace output;

namespace io {
    void setIn(string s) { freopen(s.c_str(),"r",stdin); }
    void setOut(string s) { freopen(s.c_str(),"w",stdout); }
    void setIO(string s = "") {
        ios_base::sync_with_stdio(0); cin.tie(0); // fast I/O
        if (sz(s)) { setIn(s+".in"), setOut(s+".out"); } // for USACO
    }
}

using namespace io;

template<class T> T invGeneral(T a, T b) {
    a %= b; if (a == 0) return b == 1 ? 0 : -1;
    T x = invGeneral(b,a);
    return x == -1 ? -1 : ((1-(ll)b*x)/a+b)%b;
}

template<class T> struct modular {
    T val;
    explicit operator T() const { return val; }
    modular() { val = 0; }
    template<class U> modular(const U& v) {
        val = (-MOD <= v && v <= MOD) ? v : v % MOD;
        if (val < 0) val += MOD;
    }
    friend ostream& operator<<(ostream& os, const modular& a) { return os << a.val; }
    friend bool operator==(const modular& a, const modular& b) { return a.val == b.val; }
    friend bool operator!=(const modular& a, const modular& b) { return !(a == b); }

    modular operator-() const { return modular(-val); }
    modular& operator+=(const modular& m) { if ((val += m.val) >= MOD) val -= MOD; return *this; }
    modular& operator-=(const modular& m) { if ((val -= m.val) < 0) val += MOD; return *this; }
    modular& operator*=(const modular& m) { val = (ll)val*m.val%MOD; return *this; }
    friend modular exp(modular a, ll p) {
        modular ans = 1; for (; p; p /= 2, a *= a) if (p&1) ans *= a;
        return ans;
    }
    friend modular inv(const modular& a) { return invGeneral(a.val,MOD); }
    // inv is equivalent to return exp(b,b.mod-2) if prime
    modular& operator/=(const modular& m) { return (*this) *= inv(m); }

    friend modular operator+(modular a, const modular& b) { return a += b; }
    friend modular operator-(modular a, const modular& b) { return a -= b; }
    friend modular operator*(modular a, const modular& b) { return a *= b; }

    friend modular operator/(modular a, const modular& b) { return a /= b; }
};

typedef modular<int> mi;
typedef pair<mi,mi> pmi;
typedef vector<mi> vmi;
typedef vector<pmi> vpmi;

int W;
ll ans[7];
```

```
void solve(int caseNum) {
    cout << 210 << endl;
    ll ring; re(ring);
    FOR(i,4,7) ans[i] = (ring>>(210/i))&((1<<7)-1);
    // cout << ring << " " << ans[6] << endl;
    cout << 50 << endl;
    re(ring);
    FOR(i,4,7) ring -= (ans[i]<<(50/i));
    FOR(i,1,4) ans[i] = (ring>>(50/i))&((1<<7)-1);
    FOR(i,1,7) cout << ans[i] << ' ';
    cout << endl;
    int verdict; re(verdict);
    // cerr << "Solved #" << caseNum << "\n";
}

int main() {
    int T; re(T,W);
    FOR(i,1,T+1) {
        // pr("Case #",i,": ");
        solve(i);
    }
}

/* stuff you should look for
    * int overflow, array bounds
    * special cases (n=1?), set tle
    * do smth instead of nothing and stay organized
*/
```