```
//Alexandru Vasiluta

#include <bits/stdc++.h>
using namespace std;
string s;
vector<int> urmator = {
    0, 8, 8, 8, 8, 8, 8,            // 0-6
    13, 13, 13, 13, 13,             // 7-11
    18, 18, 18, 18, 18,             // 12-16
    23, 23, 23, 23, 23,             // 17-21
    28, 28, 28, 28, 28,             // 22-26
    36, 36, 36, 36, 36, 36, 36, 36, // 27-34
    41, 41, 41, 41, 41,             // 35-39
    46, 46, 46, 46, 46,             // 40-44
    51, 51, 51, 51, 51,             // 45-49
    -1, -1, -1,                     // 50-52
};
vector<int> prime = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41,
43, 47, 53};
vector<int> pp = {9, 16, 25, 36, 49, 64};

bool occupied(int newPoz, vector<int> op, vector<int> pl)
{
    if (newPoz > 52)
        return true;
    for (const auto &i : op)
        if (i == newPoz)
            return true;
    for (const auto &i : pl)
        if (i == newPoz)
            return true;
    return false;
}

int main()
{
    while (!cin.eof())
    {
        vector<int> op(3), pl(3);
        int n;
        cin >> op[0] >> op[1] >> op[2] >> pl[0] >> pl[1] >> pl[2] >>
n;
        if (cin.eof())
            break;
        sort(op.begin(), op.end());
        sort(pl.begin(), pl.end());
        for (int i = 0; i < n; i++)
        {
            int d;
            cin >> d;
            int newPoz = pl[0] + d, oldPoz = pl[0];
            pl[0] = 9999;
            int ok = 1;
            if (newPoz > 52)
                ok = 0;
            for (const auto &p : op)
```

```
    if (p == newPoz)
        ok = 0;
for (const auto &p : pl)
    if (p == newPoz)
        ok = 0;
if (!ok)
{
    pl[0] = oldPoz;
    continue;
}
if (newPoz == 52)
{
    vector<int> newPl;
    newPl.push_back(pl[1]);
    newPl.push_back(pl[2]);
    pl = newPl;
    sort(pl.begin(), pl.end());
    continue;
}
if (binary_search(prime.begin(), prime.end(), newPoz))
{
    for (int i = 1; i <= 6; i++, newPoz++)
        if (occupied(newPoz + 1, op, pl))
            break;
    if (newPoz == 52)
    {
        vector<int> newPl;
        newPl.push_back(pl[1]);
        newPl.push_back(pl[2]);
        pl = newPl;
        sort(pl.begin(), pl.end());
        continue;
    }
    pl[0] = newPoz;
    sort(pl.begin(), pl.end());
    continue;
}
if (binary_search(pp.begin(), pp.end(), newPoz))
{
    for (int i = 1; i <= 6; i++, newPoz--)
        if (occupied(newPoz - 1, op, pl))
            break;
    if (newPoz == 52)
    {
        vector<int> newPl;
        newPl.push_back(pl[1]);
        newPl.push_back(pl[2]);
        pl = newPl;
        sort(pl.begin(), pl.end());
        continue;
    }
    pl[0] = newPoz;
    sort(pl.begin(), pl.end());
    continue;
}
```

```
            if (urmator[oldPoz] > newPoz)
            {
                pl[0] = newPoz;
                sort(pl.begin(), pl.end());
                continue;
            }
            if (occupied(newPoz / d * d, op, pl))
            {
                pl[0] = oldPoz;
                continue;
            }

            newPoz = newPoz / d * d;
            if (newPoz == 52)
            {
                vector<int> newPl;
                newPl.push_back(pl[1]);
                newPl.push_back(pl[2]);
                pl = newPl;
                sort(pl.begin(), pl.end());
                continue;
            }
            pl[0] = newPoz;
            sort(pl.begin(), pl.end());
        }
        if (pl.size() == 0)
        {
            cout << "GAME OVER\n";
        }
        else
        {
            for (const auto &p : pl)
                cout << p << " ";
            cout << "\n";
        }
    }
    return 0;
}
```