DFS Edges

HackerRank

Let G be a connected, directed graph with vertices numbered from 1 to n such that any vertex is reachable from vertex 1. In addition, any two distinct vertices, u and v, are connected by *at most* one edge (u, v).

Consider the standard *DFS* (Depth-First Search) algorithm starting from vertex 1. As every vertex is reachable, each edge (u, v) of G is classified by the algorithm into one of four groups:

1. *tree edge*: If v was discovered for the first time when we traversed (u, v).
2. *back edge*: If v was already on the stack when we tried to traverse (u, v).
3. *forward edge*: If v was already discovered *while* u was on the stack.
4. *cross edge*: Any edge that is not a *tree*, *back*, or *forward* edge.

To better understand this, consider the following C++ pseudocode:

```
// initially false
bool discovered[n];

// initially false
bool finished[n];

vector<int> g[n];

void dfs(int u) {
    // u is on the stack now
    discovered[u] = true;
    for (int v: g[u]) {
        if (finished[v]) {
            // forward edge if u was on the stack when v was discovered
            // cross edge otherwise
            continue;
        }
        if (discovered[v]) {
            // back edge
            continue;
        }
        // tree edge
        dfs(v);
    }
    finished[u] = true;
    // u is no longer on the stack
}
```

Given four integers, t, b, f, and c, construct any graph G having exactly t *tree edges*, exactly b *back edges*, exactly f *forward edges*, and exactly c *cross edges*. Then print G according to the *Output Format* specified below.

Input Format

A single line of four space-separated integers describing the respective values of t, b, f, and c.

Constraints

- $0 \leq t, b, f, c \leq 10^5$

Output Format

If there is no such graph G, print `-1`; otherwise print the following:

1. The first line must contain an integer, n, denoting the number of vertices in G.
2. Each line i of the n subsequent lines must contain the following space-separated integers:
   - The first integer is the outdegree, d_i, of vertex i.
   - This is followed by d_i distinct numbers, $v_{i,j}$, denoting edges from u to $v_{i,j}$ for $1 \leq j \leq d_i$. The order of each $v_{i,j}$ should be the order in which a *DFS* considers edges.

Sample Input 0

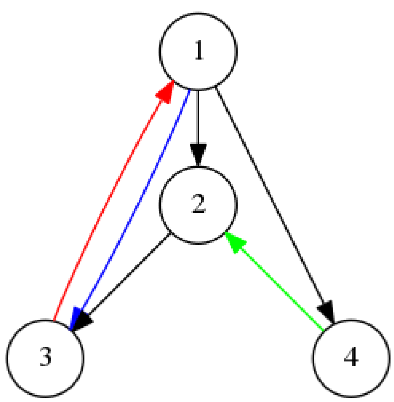
```
3 1 1 1
```

Sample Output 0

```
4
3 2 4 3
1 3
1 1
1 2
```

Explanation 0

The *DFS* traversal order is: $1, 2, 3, 2, 1, 4, 1$. Thus, $(1, 2)$, $(2, 3)$ and $(1, 4)$ are *tree edges*; $(3, 1)$ is a *back edge*; $(1, 3)$ is a *forward edge*; and $(4, 2)$ is a *cross edge*. This is demonstrated by the diagram below, in which *tree edges* are black, *forward edges* are blue, *back edges* are red, and *cross edges* are green.



Sample Input 1

```
1 10 20 30
```

Sample Output 1

```
-1
```

Explanation 1

No such graph exists satisfying the given values.