Problem C. Challenge

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Many programming competition problems on trees are solved using the *centroid decomposition*: given a tree, we first find a *centroid* — such vertex that after removing it from the tree the remaining pieces all have at most half the number of vertices of the original tree. Then, we remove the found centroid from the tree and recursively apply the process to each remaining piece. Since the size of each piece decreases at least 2x on each level, there are at most $log_2(n)+1$ levels for a tree with n vertices (in other words, for any vertex there are at most $log_2(n)+1$ pieces containing this vertex that are processed at any point of the algorithm).

You have noticed that another contestant has made a bug in their centroid decomposition: instead of finding a centroid, they find a *center* of the tree at each stage — a vertex that minimizes the maximum distance to other vertices in the tree. More formally, if d_{ij} is the distance (in tree edges) between vertices i and j, the value $r = min_i(max_j(d_{ij}))$ is called the *radius* of the tree, and any vertex k such that $max_j(d_{kj}) = r$ is called a center of the tree.

It turns out that there can be more than $log_2(n) + 1$ levels of decomposition because of this bug. In order to challenge this solution, you need to construct a tree with at most n vertices for which this 'center decomposition' will have at least $\lfloor\sqrt{n}\rfloor$ levels (square root of n rounded down). More precisely, there must be at least one vertex which belongs to at least $\lfloor\sqrt{n}\rfloor$ different subsets processed at some point of the decomposition.

You can assume that whenever a tree at some stage has multiple centers, the decomposition picks the one with the smallest vertex number as the center to remove from the tree.

Input

The only line of the input contains the integer n, $1 \le n \le 100000$.

Output

On the first line of output, print an integer m ($1 \le m \le n$) denoting the number of vertices in your tree. On the next $m - 1$ lines describe the edges of the tree. Each edge should be described by the two numbers of vertices it connects. The vertices are numbered from 1 to m.

At least one vertex of the tree must belong to at least $\lfloor\sqrt{n}\rfloor$ different pieces in the process that picks the center of the tree (the one with the smallest vertex number if there are several), removes it from the tree, and recursively applies itself to all remaining pieces.

Example

standard input	standard output
4	3 1 2 2 3