A. Suborrays

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

A permutation of length n is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (2 appears twice in the array) and $[1, 3, 4]$ is also not a permutation ($n = 3$ but there is 4 in the array).

For a positive integer n, we call a permutation p of length n **good** if the following condition holds for every pair i and j ($1 \le i \le j \le n$) —

- $(p_i \text{ OR } p_{i+1} \text{ OR } \ldots \text{ OR } p_{j-1} \text{ OR } p_j) \ge j - i + 1$, where OR denotes the bitwise OR operation.

In other words, a permutation p is **good** if for every subarray of p, the OR of all elements in it is not less than the number of elements in that subarray.

Given a positive integer n, output any **good** permutation of length n. We can show that for the given constraints such a permutation always exists.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). Description of the test cases follows.

The first and only line of every test case contains a single integer n ($1 \le n \le 100$).

Output

For every test, output any **good** permutation of length n on a separate line.

Example

input
3 1 3 7
output
1 3 1 2 4 3 5 2 7 1 6

Note

For $n = 3$, $[3, 1, 2]$ is a good permutation. Some of the subarrays are listed below.

- $3 \text{ OR } 1 = 3 \ge 2$ ($i = 1, j = 2$)
- $3 \text{ OR } 1 \text{ OR } 2 = 3 \ge 3$ ($i = 1, j = 3$)
- $1 \text{ OR } 2 = 3 \ge 2$ ($i = 2, j = 3$)
- $1 \ge 1$ ($i = 2, j = 2$)

Similarly, you can verify that $[4, 3, 5, 2, 7, 1, 6]$ is also good.