**Maximum OR**

Given a 0-indexed integer array nums of length n and an integer k, you are allowed to perform an operation on any element by multiplying it by 2. The goal is to find the maximum possible value of the bitwise OR operation performed on all elements in the nums array after applying the operation at most k times.

Note that the bitwise OR operation (|) between two integers a and b results in a new integer where each bit is set if it is set in either a or b.

**Important Note:** *Ensure that you save your solution before progressing to the next question and before submitting your answer.*

**Example**

INPUT [12,9] 1 OUTPUT 30

**Exercise-1**

Input :

12 9

2

Output :

57

**Exercise-2**

Input:

3 2 4

2

Output:

19

import java.util.ArrayList;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

ArrayList<Integer> ary = new ArrayList<>();

while (input.hasNextInt()) {

ary.add(input.nextInt());

}

int n = ary.size();

int a = ary.get(n - 1);

ary.remove(n - 1);

int[] arr = new int[n - 1];

for (int i = 0; i < ary.size(); i++) {

arr[i] = ary.get(i);

}

System.out.println(maxBitwiseOr(arr,a));

}

public static long maxBitwiseOr(int[] nums, int k) {

long currentOr = 0;

for (int num : nums) {

currentOr |= num;

}

long maxOr = currentOr;

for (int i = 0; i < nums.length; i++) {

long newOr = currentOr;

newOr ^= nums[i];

long tempNum = nums[i];

for (int j = 0; j < k; j++) {

tempNum \*= 2;

newOr |= tempNum;

maxOr = Math.max(maxOr, newOr);

}

}

return maxOr;

}

}