

## **Maximum Product**

Given the array of integers `nums`, you will choose two different indices `i` and `j` of that array. *Return the maximum value of  $(\text{nums}[i]-1) * (\text{nums}[j]-1)$ .*

### **Input Format:**

Vector `nums[]` of integers is passed in the function.

### **Output Format:**

Return a single integer as required.

### **Constraints:**

- $2 \leq \text{nums.length} \leq 500$
- $1 \leq \text{nums}[i] \leq 10^3$

### **Sample Input**

```
nums = [3,4,5,2]
```

### **Sample Output**

12

### **Explanation**

$(\text{nums}[1]-1) * (\text{nums}[2]-1) = (4-1) * (5-1) = 12$

**Solution:** `maxProduct.cpp`

## **Reduced array size to half**

You are given an integer array `arr`. You can choose a set of integers and remove all the occurrences of these integers in the array.

Return the minimum size of the set so that at least half of the integers of the array are removed.

### **Constraints:**

- $1 \leq \text{arr.length} \leq 10^5$
- $1 \leq \text{arr}[i] \leq 10^5$

### **Input Format:**

Vector `arr[]` is passed in the given function.

**Output Format:**

Return a single integer as required.

**Sample Test case:**

**Input:** `arr = [3,3,3,3,5,5,5,2,2,7]`

**Output:** 2

**Explanation:**

Choosing `{3,7}` will make the new array `[5,5,5,2,2]` which has size 5 (i.e equal to half of the size of the old array).

Possible sets of size 2 are `{3,5}`, `{3,2}`, `{5,2}`.

Choosing set `{2,7}` is not possible as it will make the new array `[3,3,3,3,5,5,5]` which has size greater than half of the size of the old array.

**Solution:** `reduceToHalf.cpp`

## **Weakest Rows**

You are given an  $m \times n$  binary matrix `mat` of 1's (representing soldiers) and 0's (representing civilians). The soldiers are positioned in front of the civilians. That is, all the 1's will appear to the left of all the 0's in each row.

A row  $i$  is weaker than a row  $j$  if one of the following is true:

- The number of soldiers in row  $i$  is less than the number of soldiers in row  $j$ .
- Both rows have the same number of soldiers and  $i < j$ .

Return the indices of the  $k$  weakest rows in the matrix ordered from weakest to strongest.

**Input Format:**

Integer  $m, n$ , matrix `mat` and  $k$  are passed in the function.

**Output Format:**

Return the vector as required.

**Constraints:**

- $m == \text{mat.length}$
- $n == \text{mat}[i].\text{length}$
- $2 \leq n, m \leq 100$
- $1 \leq k \leq m$

matrix[i][j] is either 0 or 1.

### Sample Input

m = 4 n = 4 [[1,0,0,0], [1,1,1,1], [1,0,0,0], [1,0,0,0]], k = 2

### Sample Output

[0,2]

### Explanation

weakness of the rows is in the order- 0, 2, 3, 1

**Solution:** weakestRows.cpp

## Relative Ranks

You are given an integer array score of size n, where score[i] is the score of the ith athlete in a competition. All the scores are guaranteed to be unique.

The athletes are placed based on their scores, where the 1st place athlete has the highest score, the 2nd place athlete has the 2nd highest score, and so on. The placement of each athlete determines their rank:

- The 1st place athlete's rank is "Gold Medal".
- The 2nd place athlete's rank is "Silver Medal".
- The 3rd place athlete's rank is "Bronze Medal".
- For the 4th place to the nth place athlete, their rank is their placement number (i.e., the xth place athlete's rank is "x").

Return an array answer of size n where answer[i] is the rank of the ith athlete.

### Input Format:

An array score[] is passed in the given function.

### Output Format:

Return an array of ranks of all athletes.

### Constraints:

- n == score.length
- 1 <= n <= 10<sup>4</sup>
- 0 <= score[i] <= 10<sup>6</sup>

- All the values in score are unique.

**Sample Test Cases:**

**Input 1:**

**score** = [5,4,3,2,1]

**Output 1:**

**["Gold Medal", "Silver Medal", "Bronze Medal", "4", "5"]**

**Solution:** relativeRanks.cpp