# Dynamic Array



- Declare a 2-dimensional array, $arr$, of $n$ empty arrays. All arrays are zero indexed.
- Declare an integer, $lastAnswer$, and initialize it to $0$.
- There are $2$ types of queries, given as an array of strings for you to parse:
  1. Query: `1 x y`
     1. Let $idx = ( (x \oplus lastAnswer) \% n )$.
     2. Append the integer $y$ to $arr[idx]$.
  2. Query: `2 x y`
     1. Let $idx = ( (x \oplus lastAnswer) \% n )$.
     2. Assign the value $arr[idx][y \% size(arr[idx])]$ to $lastAnswer$.
     3. Store the new value of $lastAnswer$ to an answers array.

**Note:** $\oplus$ is the bitwise XOR operation, which corresponds to the `^` operator in most languages. Learn more about it on Wikipedia. $\%$ is the modulo operator.

Finally, size(arr[idx]) is the number of elements in arr[idx]

**Function Description**

Complete the dynamicArray function below.

dynamicArray has the following parameters:

- int n: the number of empty arrays to initialize in $arr$

- string queries[q]: query strings that contain 3 space-separated integers

**Returns**

- int[]: the results of each type 2 query in the order they are presented

**Input Format**

The first line contains two space-separated integers, $n$, the size of $arr$ to create, and $q$, the number of queries, respectively.

Each of the $q$ subsequent lines contains a query string, $queries[i]$.

**Constraints**

- $1 \leq n, q \leq 10^5$
- $0 \leq x, y \leq 10^9$
- It is guaranteed that query type $2$ will never query an empty array or index.

**Sample Input**

```
2 5
1 0 5
1 1 7
1 0 3
2 1 0
2 1 1
```

**Sample Output**

```
7
3
```

**Explanation**

Initial Values:

$n = 2$

$lastAnswer = 0$

Author ikbalkazar

Difficulty Easy

Max Score 100

Submitted By 3983

NEED HELP?

View discussions

View editorial

View top submissions

RATE THIS CHALLENGE

MORE DETAILS

Download problem statement

Download sample test cases

Suggest Edits

$arr[0]$ = [ ]

$arr[1]$ = [ ]

Query 0: Append $5$ to $arr[(\ (0 \oplus 0)\ \%\ 2\ )] = arr[0]$.

$lastAnswer = 0$

$arr[0]$ = [5]

$arr[1]$ = [ ]

Query 1: Append $7$ to $arr[(\ (1 \oplus 0)\ \%\ 2\ )] = arr[1]$.

$arr[0]$ = [5]

$arr[1]$ = [7]

Query 2: Append $3$ to $arr[(\ (0 \oplus 0)\ \%\ 2\ )] = arr[0]$.

$lastAnswer = 0$

$arr[0]$ = [5, 3]

$arr[1]$ = [7]

Query 3: Assign the value at index $0$ of $arr[(\ (1 \oplus 0)\ \%\ 2\ )] = arr[1]$ to $lastAnswer$, print $lastAnswer$.

$lastAnswer = 7$

$arr[0]$ = [5, 3]

$arr[1]$ = [7]

```
7
```

Query 4: Assign the value at index $1$ of $arr[(\ (1 \oplus 7)\ \%\ 2\ )] = arr[0]$ to $lastAnswer$, print $lastAnswer$.

$lastAnswer = 3$

$arr[0]$ = [5, 3]

$arr[1]$ = [7]

```
3
```

Change Theme Language JavaScript (Node.js)

```
'use strict';

const fs = require('fs');

process.stdin.resume();
process.stdin.setEncoding('utf-8');

let inputString = '';
let currentLine = 0;

process.stdin.on('data', function(inputStdin) {
    inputString += inputStdin;
});

process.stdin.on('end', function() {
    inputString = inputString.split('\n');

    main();
});

function readLine() {
    return inputString[currentLine++];
}

/*
 * Complete the 'dynamicArray' function below.
 *
```

```
28    * The function is expected to return an INTEGER_ARRAY.
29    * The function accepts following parameters:
```

Line: 60 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code