

PRACTICE

COMPETE

**JOBS** 

**LEADERBOARD** 





pruthvishalcodi1 >

All Contests > ALCoding Summer Weekly Contest 3 > AND xor OR

# AND xor OR



Problem

Submissions

Leaderboard

Discussions

Editorial

Given an array  $A \parallel$  of N distinct elements. Let  $M_1$  and  $M_2$  be the smallest and the next smallest element in the interval [L,R]where  $1 \leq L < R \leq N$ .

$$S_i = (((M_1 \wedge M_2) \oplus (M_1 \vee M_2)) \wedge (M_1 \oplus M_2)).$$

where  $\land$ ,  $\lor$ ,  $\oplus$ , are the bitwise operators AND, OR and XOR respectively.

Your task is to find the maximum possible value of  $S_i$ .

#### **Input Format**

First line contains integer N.

Second line contains N integers, representing elements of the array A[].

#### Constraints

 $1 < N \le 10^6$ 

 $1 \leq A_i \leq 10^9$ 

### **Output Format**

Print the value of maximum possible value of  $S_i$ .

## Sample Input

9 6 3 5 2

Sample Output

15

#### **Explanation**

Consider the interval [1,2] the result will be maximum.

$$(((9 \land 6) \oplus (9 \lor 6)) \land (9 \oplus 6)) = 15$$

Submissions: 36 Max Score: 30

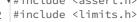
Rate This Challenge:

More

Current Buffer (saved locally, editable) & 49



1 ▼#include <assert.h>





C





```
3 #include <math.h>
   #include <stdbool.h>
  #include <stdio.h>
  #include <stdlib.h>
6
7 #include <string.h>
9 char* readline();
10 char** split_string(char*);
11
12 ▼/*
13
    * Complete the andXorOr function below.
14
15 vint andXorOr(int a_count, int* a) {
16 ▼
        * Write your code here.
17
18
19
20
21
22 int main()
23 ▼ {
24
        FILE* fptr = fopen(getenv("OUTPUT_PATH"), "w");
25
26
        char* a_count_endptr;
27
        char* a_count_str = readline();
28
        int a_count = strtol(a_count_str, &a_count_endptr, 10);
29
        if (a_count_endptr == a_count_str || *a_count_endptr != '\0') { exit(EXIT_FAILURE); }
30 ▼
31
32
        char** a_temp = split_string(readline());
33
34
        int a[a_count];
35
36 ▼
        for (int a_itr = 0; a_itr < a_count; a_itr++) {</pre>
37
            char* a_item_endptr;
            char* a_item_str = a_temp[a_itr];
38 ▼
            int a_item = strtol(a_item_str, &a_item_endptr, 10);
39
40
            if (a_item_endptr == a_item_str || *a_item_endptr != '\0') { exit(EXIT_FAILURE); }
41
42
            a[a_itr] = a_item;
43 1
44
45
        int result = andXorOr(a_count, a);
46
47
48
        fprintf(fptr, "%d\n", result);
49
50
        fclose(fptr);
51
52
        return 0;
   }
53
54
55 ▼char* readline() {
56
        size_t alloc_length = 1024;
        size_t data_length = 0;
57
58
        char* data = malloc(alloc_length);
59
60 ▼
        while (true) {
            char* cursor = data + data_length;
61
            char* line = fgets(cursor, alloc_length - data_length, stdin);
62
63
            if (!line) { break; }
64 ▼
65
            data_length += strlen(cursor);
66
67
            if (data_length < alloc_length - 1 || data[data_length - 1] == '\n') { break; }</pre>
68 •
69
70
            size_t new_length = alloc_length << 1;</pre>
71
            data = realloc(data, new_length);
72
73 🔻
            if (!data) { break; }
74
75
            alloc_length = new_length;
76
```

```
77
         if (data[data_length - 1] == '\n') {
    data[data_length - 1] = '\0';
 78 ▼
 79 ▼
80
81
82
         data = realloc(data, data_length);
83
84
         return data;
85
    }
86
87 ▼char** split_string(char* str) {
         char** splits = NULL;
88
         char* token = strtok(str, " ");
89
90
91
         int spaces = 0;
92
93 🔻
         while (token) {
              splits = realloc(splits, sizeof(char*) * ++spaces);
94
95 ▼
              if (!splits) {
96
                  return splits;
97
98
              splits[spaces - 1] = token;
99 ▼
100
              token = strtok(NULL, " ");
101
102
103
         return splits;
104
105 }
106
                                                                                                       Line: 1 Col: 1
```

<u>♣ Upload Code as File</u> Test against custom input

Run Code

Submit Code

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature