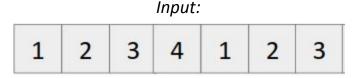
Bit Manipulation Challenges

Challenge 1

Write a program to find a unique number in an array where all numbers except one, are present twice.

Hint: $A \oplus B \oplus B \oplus A \oplus C = C$. All those numbers which occur twice will get nullified after \oplus operation.

Sample Test Case:



Output: 4

Code

```
#include<iostream>
using namespace std;
int unique(int arr[], int n) {
    int xorsum = 0;
    for (int i = 0; i < n; i++) {
        xorsum = xorsum ^ arr[i];
    }
    return xorsum;
}
int main() {
    int arr[] = {1, 2, 3, 4, 1, 2, 3};
    cout << unique(arr, 7) << endl;
    return 0;
}</pre>
```

Apni Kaksha

Challenge 2

Q2. Write a program to find 2 unique numbers in an array where all numbers except two, are present twice.

Logic

- 1. Take XOR of all the elements and let that xor value be x. All the repeating elements will get nullified and xor of only two unique elements will last. (as $a \oplus a = 0$).
- 2. There will be at least one bit set in x. Using that set bit, divide the original set of numbers into two sets
 - a. First set which contains all the elements with that bit set.
 - b. Second set which contains all the elements with that bit unset.
- 3. Take xor of both the sets individually, and let those xor values be x1 and x2.
- 4. Voila, x1 and x2 are our unique numbers. Reason: both the above sets contain one of the unique elements and rest elements of the sets occur twice which will get nullified after ⊕ operation.

Sample Test Case:

Input:



Output: 5 7

Apmi Kaksha

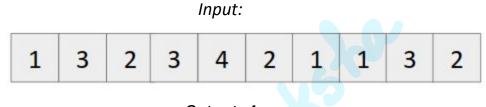
Code

```
#include<iostream>
using namespace std;
int setBit(int n, int pos) {
    return ((n & (1 << pos)) != 0);
void unique(int arr[], int n) {
   int xorsum = 0;
        xorsum = xorsum ^ arr[i];
   int tempxor = xorsum;
    int setbit = 0;
   int pos = 0;
   while (setbit != 1) {
       setbit = xorsum & 1;
       pos++;
       xorsum = xorsum >> 1;
    int newxor = 0;
        if (setBit(arr[i], pos - 1)) {
           newxor = newxor ^ arr[i];
    cout << newxor << endl;</pre>
    cout << (tempxor ^ newxor) << endl;</pre>
int main() {
   unique(arr, 8);
```

Challenge 3

Q3. Write a program to find a unique number in an array where all numbers except one, are present thrice.

Sample Test Case:



Output: 4

Logic

1. We will maintain an array of 64 size which will store the number of times ith bit has occurred in the array.

3	6 4	0 0	0 0	0 0	0	64 bits
---	-----	-----	-----	-----	---	---------

- 2. Take the modulo of each element of this array with 3. Resultant array will represent the binary representation of the unique number.
- 3. Convert that binary number to decimal number and output it.

Apni Kaksha

Code

```
#include<iostream>
using namespace std;
bool getBit(int n, int pos) {
    return ((n & (1 << pos)) != 0);
int setBit( int n, int pos) {
    return (n | (1 << pos));
int unique(int arr[], int n) {
    int result = 0;
    for (int i = 0; i < 64; i++) {
            if (getBit(arr[j], i)) {
                sum++;
        if (sum % 3 != 0) {
            result = setBit(result, i);
    return result;
int main() {
    cout << unique(arr, 10) << endl;</pre>
```