1. **Write a program to find the binary representation of a given number.**

Sample Input: num = 4

Sample Output: 100

Explanation: The Binary Representation of 4 is 100

**Code:**

*//21161 Shaik Nazeer CSE-B*

#include<bits/stdc++.h>

#define ll *long* *long*

#define loop(*i*,*n*) for(*int* i = 0; i < n; i++)

#define loop1(*i*,*n*) for(*int* i = 1; i <= n; i++)

using *namespace* std;

*int* main()

{

*int* n;

    cin>>n;

    string s,temp;

    while(n) {

        temp = n%2+'0';

        s = temp+s;

        n/=2;

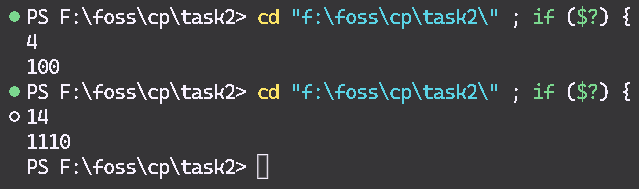
    }

    cout<<s<<endl;

    return 0;

}

**Screenshot:**



1. **Given an array of size N-1 such that it only contains distinct integers in the range of 1 to N. Find the missing element.**

Sample Input: N = 10

A[] = {6,1,2,9,3,4,7,10,5}

Sample Output: 8

Explanation: The Missing Number is 8 in the array.

**Code:**

//21161 Shaik Nazeer CSE-B

#include<bits/stdc++.h>

#define ll long long

#define loop(i,n) for(int i = 0; i < n; i++)

#define loop1(i,n) for(int i = 1; i <= n; i++)

using namespace std;

int main()

{

    int n;

    cin>>n;

    int a[n];

    loop(i,n) cin>>a[i];

    sort(a,a+n);

    loop(i,n) {

        if(a[i]!=i+1) {

            cout<<i+1<<endl;

            return 0;

        }

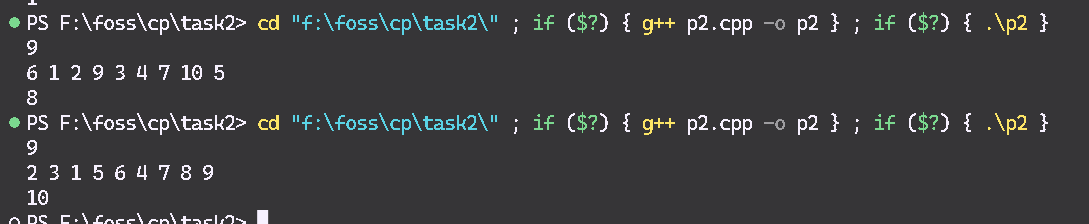
    }

    cout<<n+1<<endl;

    return 0;

}

**Screenshot:**

****

1. **Find the first set bit for a given number.**

Sample Input: num = 12

Sample Output: 3

Explanation: 1100 is the binary representation of the given number 12. Set bit is nothing but the ‘1’ bits that are present in a binary number. In 1100 the first set bit from the right occurs in third position. Thus, the output is 3.

**Code:**

*//21161 Shaik Nazeer CSE-B*

#include<bits/stdc++.h>

#define ll *long* *long*

#define loop(*i*,*n*) for(*int* i = 0; i < n; i++)

#define loop1(*i*,*n*) for(*int* i = 1; i <= n; i++)

using *namespace* std;

*int* main()

{

*int* n,bitmask=1,pos=1;

    cin>>n;

    while(true) {

        if(n&bitmask){

            cout<<pos<<endl;

            break;

        }

        pos++;

        n = n>>1;

    }

    return 0;

}

**Screenshot:**

****

1. **Find the two numbers with odd occurrences in an unsorted array.**

Sample Input: {2, 4 , 2, 5, 7, 5, 4, 6, 5, 7}

Output: 5 and 6

Explanation: The element 5 and 6 occurs odd times in the array that is element 5 occurs three times while element 6 occurs one time. Thus, the output is 5 and 6

Input Constraints: The array should contain only two elements that occurs odd no of times and not more than two.

Other Constraints: You cannot use nested loops to solve this problem

**Code:**

*//21161 Shaik Nazeer CSE-B*

#include<bits/stdc++.h>

#define ll *long* *long*

#define loop(*i*,*n*) for(*int* i = 0; i < n; i++)

#define loop1(*i*,*n*) for(*int* i = 1; i <= n; i++)

using *namespace* std;

*int* main()

{

*int* n,x=-1,y=-1;

    cin>>n;

*int* a[n];

    loop(i,n) cin>>a[i];

    sort(a,a+n);

    loop(i,n){

        if(a[i]==a[i+1]){

            i++;

        }else{

            if(x==-1) x=a[i];

            else if(y==-1) y=a[i];

        }

        if(x!=-1 && y!=-1){

            cout<<x<<" "<<y<<endl;

            break;

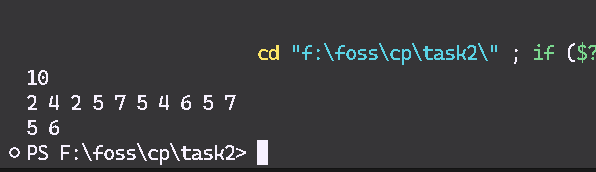
        }

    }

    return 0;

}

**Screenshot:**

****

1. **Count the no of set bits i.e no of ‘1’ bits in an integer.**

Sample Input: 7

Sample Output: 3

Explanation: The binary representation of 7 is 111 and the no of ‘1’ bits present in 7 is three. Thus, the output is 3.

**Code:**

*//21161 Shaik Nazeer CSE-B*

#include<bits/stdc++.h>

#define ll *long* *long*

#define loop(*i*,*n*) for(*int* i = 0; i < n; i++)

#define loop1(*i*,*n*) for(*int* i = 1; i <= n; i++)

using *namespace* std;

*int* main()

{

*int* n,bitmask=1,cnt=0;

    cin>>n;

    while(n) {

        if(n&bitmask){

            cnt++;

        }

        n = n>>1;

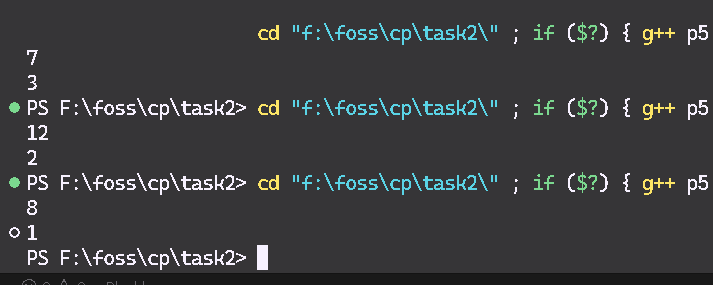
    }

    cout<<cnt<<endl;

    return 0;

}

**Screenshot:**

****