**Rotate Bits: -**

Basic Accuracy: 20.8% Submissions: 38K+ Points: 1

Given an integer **N** and an integer **D**, rotate the binary representation of the integer N by Ddigits to the **left**as well as **right**and return the results in their **decimal representation**after each of the rotation.  
**Note**: Integer N is stored using **16 bits**. i.e. 12 will be stored as 0000000000001100.

**Example 1:**

**Input:**

N = 28, D = 2

**Output:**

112

7

**Explanation**:   
28 in Binary is: 0000000000011100

Rotating left by 2 positions, it becomes 0000000001110000 = 112 (in decimal).  
Rotating right by 2 positions, it becomes 0000000000000111 = 7 (in decimal).

**Example 2:**

**Input**:

N = 29, D = 2

**Output:**

116

16391

**Explanation**:   
29 in Binary is: 0000000000011101

Rotating left by 2 positions, it becomes 0000000001110100 = 116 (in decimal).

Rotating right by 2 positions, it becomes 0000000010000111 = 16391 (in decimal).

**Your Task:**  
You don't need to read input or print anything. Your task is to complete the function **rotate()**which takes the integer N and integer D as inputs and returns an array of size 2 where arr[0] = Decimal value after left rotation and arr[1] = Decimal value after right rotation.

**Expected Time Complexity:**O(1).  
**Expected Auxiliary Space:**O(1).

**Constraints:**  
1 <= N <  216  
1 <= D <= 105

**Code: -**

//{ Driver Code Starts

#include<bits/stdc++.h>

using namespace std;

// } Driver Code Ends

class Solution

{

public:

vector <int> rotate (int n, int d){

d = d % 16;

unsigned short int num = n;

unsigned short int left = (num >> (16 - d)) | (num << d);

unsigned short int right = (num << (16 - d)) | (num >> d);

return {left, right};

}

};

//{ Driver Code Starts.

int main()

{

int t; cin >> t;

while (t--)

{

int n, d; cin >> n >> d;

Solution ob;

vector <int> res = ob.rotate (n, d);

cout << res[0] << endl << res[1] << endl;

}

}

// Contributed By: Pranay Bansal

// } Driver Code Ends

**T.C: - O(1)**

**S.C: - O(1)**