10th Jan 2023

Bit Manipulation : : Easy

Given a 32 bit unsigned integer **num** and an integer **i**. Perform following operations on the number -

1. **Get** ith bit

2. **Set** ith bit

3. **Clear** ith bit

Note : For better understanding, we are starting bits from 1 instead 0. (1-based)

Example 1:

```
Input: 70 3
Output: 1 70 66
Explanation: Bit at the 3rd position from LSB is 0. (1 0 0 0 1 1 0)
The value of the given number after setting the 3rd bit is 70.
The value of the given number after clearing 3rd bit is 66. (1 0 0 0 0 1 0)
```

Example 2:

```
Input: 8 1
Output: 0 9 8
Explanation: Bit at the first position from LSB is 0. (1 0 0 0)
The value of the given number after setting the 3rd bit is 9. (1 0 0 1)
The value of the given number after clearing 3rd bit is 66. (1 0 0 0)
```

Your Task:

Complete the function **bitManipulation()** which takes two integers num and i as input and prints the results after each operation separted by a space in the same line. You don't have to print any new lines after printing the output, that will be handled by driver code.

Constraints:

$0<=num<=10^9$

$1<=i<=32$

CODE SECTION:-

```
void clearbit(int n,int i){
    int musk=1<<i-1;
    musk = ~musk;

    n=n&musk;
    cout<<n<<" ";

}
void setbit(int n,int i){

    int musk = 1<<i-1;
    n= n | musk;
    cout<<n<<" ";

}
void getbit(int n,int i){

    if(n>>i-1 & 1==1){
        cout<<"1 ";
    }
    else{
        cout<<"0 ";
    }

}
```

```
void bitManipulation(int num, int i) {
    // your code here

    getbit(num,i);
    setbit(num,i);
    clearbit(num,i);


}
```

SECOND :-

Check whether K-th bit is set or not :: Easy

Given a number **N** and a bit number **K**, check if **Kth** bit of N is set or not. A bit is called set if it is 1. Position of set bit '1' should be indexed starting with 0 from LSB side in binary representation of the number.

Example 1:

```
Input: N = 4, K = 0
Output: No
Explanation: Binary representation of 4 is 100,
in which 0th bit from LSB is not set.
So, return false.
```

Example 2:

```
Input: N = 4, K = 2
Output: Yes
Explanation: Binary representation of 4 is 100,
in which 2nd bit from LSB is set.
So, return true.
```

Example 3:

```
Input: N = 500, K = 3
Output: No
```

Explanation: Binary representation of 500 is 111110100, in which 3rd bit from LSB is not set. So, return false.

Your task:
You don't have to read input or print anything. Your task is to complete the function **checkKthbit** that takes **n** and **k** as parameters and returns either true (if kth bit is set) or false(if kth bit is not set).

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

Constraints:
$1 \leq N \leq 10^9$
$0 \leq K \leq \text{floor}(\log_2(N) + 1)$

CODE SECTION:-

```
bool checkKthBit(int n, int k)
 {
     // Your code here
     // It can be a one liner logic!! Think of it!!


     if(n>>k & 1 ==1 ){
         return true;
     }
     else{
         return false;
     }
 }
```

-: DONE FOR THE DAY :-