Problem 3064: Guess the Number Using Bitwise Questions I

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

There is a number

n

that you have to find.

There is also a pre-defined API

int commonSetBits(int num)

, which returns the number of bits where both

n

and

num

are

1

in that position of their binary representation. In other words, it returns the number of

set bits

in

n & num

, where

&

is the bitwise

AND

operator.

Return

the number

n

.

Example 1:

Input:

n = 31

Output:

31

Explanation:

It can be proven that it's possible to find

31

using the provided API.

Example 2:

Input:

n = 33

Output:

33

Explanation:

It can be proven that it's possible to find

33

using the provided API.

Constraints:

1 <= n <= 2

30

- 1

0 <= num <= 2

30

- 1

If you ask for some

num

out of the given range, the output wouldn't be reliable.

Code Snippets

C++:

```
/**
* Definition of commonSetBits API.
* int commonSetBits(int num);
*/

class Solution {
public:
int findNumber() {

}
};
```

Java:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem).
* int commonSetBits(int num);
*/

public class Solution extends Problem {
public int findNumber() {

}
}
```

Python3:

```
# Definition of commonSetBits API.
# def commonSetBits(num: int) -> int:

class Solution:
def findNumber(self) -> int:
```

Python:

```
# Definition of commonSetBits API.
# def commonSetBits(num):
```

```
# """
# :type num: int
# :rtype: int
# """

class Solution(object):
def findNumber(self):
"""
:rtype: int
"""
```

JavaScript:

```
/**
* Definition of commonSetBits API.
* @param {number} num
* @return {integer}
* var commonSetBits = function(num) {}
*/

/**
* @return {number}
*/
var findNumber = function() {

};
```

TypeScript:

```
/**
* Definition of commonSetBits API.
* var commonSetBits = function(num: number): number {}
*/

function findNumber(): number {

};
```

C#:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem).
* int CommonSetBits(int num);
*/

public class Solution : Problem {
public int FindNumber() {

}
}
```

C:

```
/**
* Definition of commonSetBits API.
* int commonSetBits(int num);
*/

int findNumber(){

}
```

Go:

```
/**
* Definition of commonSetBits API.
* func commonSetBits(num int) int;
*/

func findNumber() int {

}
```

Kotlin:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem).
* fun commonSetBits(num: Int): Int {}
*/

class Solution : Problem() {
fun findNumber(): Int {

}

}
```

Swift:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem)
* func commonSetBits(_ num: Int) -> Int
*/

class Solution : Problem {
func findNumber() -> Int {

}
}
```

Rust:

```
/**
* Definition of commonSetBits API.
* unsafe fn common_set_bits(num: i32) -> i32 {}
*/

impl Solution {
unsafe fn find_number() -> i32 {

}
}
```

Ruby:

```
# Definition of commonSetBits API.
# @param {Integer} num
# @return {Integer}
# def common_set_bits(num)

def find_number()

end
```

PHP:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem).
* @param Integer $num
* @return Integer
* public function commonSetBits($num) : Integer
*/

class Solution extends Problem {
/**
* @return Integer
*/
function findNumber() {

}
}
```

Dart:

```
/**
* Definition of commonSetBits API.
* int commonSetBits(int num);
*/

class Solution {
int findNumber() {

}
}
```

Scala:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem)
* def commonSetBits(num: Int): Int = {}
*/

class Solution extends Problem {
def findNumber(): Int = {

}
}
```

Elixir:

```
# Definition of commonSetBits API.
# common_set_bits = fn
# num :: integer -> integer
# end
# Note that due to the limitations of the language, common_set_bits is passed
to you as an anonymous function.
# To call it, you should use the dot notation. e.g., common_set_bits.(x)

defmodule Solution do
@spec find_number(common_set_bits :: (integer -> integer)) :: integer
def find_number(common_set_bits) do

end
end
```

Erlang:

```
%% Definition of commonSetBits API.
%% -spec common_set_bits(Num :: integer()) -> integer().
%% common_set_bits(Num) ->
%% .

-spec find_number() -> integer().
find_number() ->
.
```

Racket:

```
;; Definition of commonSetBits API.
#|
(define/contract (common-set-bits num)
(-> exact-integer? exact-integer?)
)
|#

(define/contract (find-number)
(-> exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Guess the Number Using Bitwise Questions I
* Difficulty: Medium
* Tags: general
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition of commonSetBits API.
* int commonSetBits(int num);
*/

class Solution {
public:
int findNumber() {

}
};
```

Java Solution:

```
/**
* Problem: Guess the Number Using Bitwise Questions I
* Difficulty: Medium
* Tags: general
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition of commonSetBits API (defined in the parent class Problem).
* int commonSetBits(int num);
*/

public class Solution extends Problem {
public int findNumber() {
}
}
```

Python3 Solution:

```
"""
Problem: Guess the Number Using Bitwise Questions I
Difficulty: Medium
Tags: general

Approach: Optimized algorithm based on problem constraints
Time Complexity: O(n) to O(n^2) depending on approach
Space Complexity: O(1) to O(n) depending on approach
"""

# Definition of commonSetBits API.
# def commonSetBits(num: int) -> int:

class Solution:
def findNumber(self) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
# Definition of commonSetBits API.
# def commonSetBits(num):
# """
# :type num: int
# :rtype: int
# """

class Solution(object):
def findNumber(self):
"""
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Guess the Number Using Bitwise Questions I
* Difficulty: Medium
* Tags: general
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition of commonSetBits API.
* @param {number} num
* @return {integer}
* var commonSetBits = function(num) {}
*/

/**
* @return {number}
*/
var findNumber = function() {

};
```

TypeScript Solution:

```
/**
* Problem: Guess the Number Using Bitwise Questions I
* Difficulty: Medium
* Tags: general
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition of commonSetBits API.
* var commonSetBits = function(num: number): number {}
*/

function findNumber(): number {
};
```

C# Solution:

```
/*
* Problem: Guess the Number Using Bitwise Questions I
* Difficulty: Medium
* Tags: general
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition of commonSetBits API (defined in the parent class Problem).
* int CommonSetBits(int num);
*/

public class Solution : Problem {
public int FindNumber() {

}
}
```

C Solution:

```
/*
* Problem: Guess the Number Using Bitwise Questions I
* Difficulty: Medium
* Tags: general
*
* Approach: Optimized algorithm based on problem constraints
* Time Complexity: O(n) to O(n^2) depending on approach
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Definition of commonSetBits API.
* int commonSetBits(int num);
```

```
*/

int findNumber(){

}
```

Go Solution:

```
// Problem: Guess the Number Using Bitwise Questions I
// Difficulty: Medium
// Tags: general
//
// Approach: Optimized algorithm based on problem constraints
// Time Complexity: O(n) to O(n^2) depending on approach
// Space Complexity: O(1) to O(n) depending on approach

/**
* Definition of commonSetBits API.
* func commonSetBits(num int) int;
*/

func findNumber() int {

}
```

Kotlin Solution:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem).
* fun commonSetBits(num: Int): Int {}
*/

class Solution : Problem() {
fun findNumber(): Int {

}
}
```

Swift Solution:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem)
* func commonSetBits(_ num: Int) -> Int
*/

class Solution : Problem {
func findNumber() -> Int {

}
}
```

Rust Solution:

```
// Problem: Guess the Number Using Bitwise Questions I
// Difficulty: Medium
// Tags: general
//
// Approach: Optimized algorithm based on problem constraints
// Time Complexity: O(n) to O(n^2) depending on approach
// Space Complexity: O(1) to O(n) depending on approach

/**
* Definition of commonSetBits API.
* unsafe fn common_set_bits(num: i32) -> i32 {}
*/

impl Solution {
unsafe fn find_number() -> i32 {

}
}
```

Ruby Solution:

```
# Definition of commonSetBits API.
# @param {Integer} num
# @return {Integer}
# def common_set_bits(num)

def find_number()

end
```

PHP Solution:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem).
* @param Integer $num
* @return Integer
* public function commonSetBits($num) : Integer
*/

class Solution extends Problem {
/**
* @return Integer
*/
function findNumber() {

}
}
```

Dart Solution:

```
/**
* Definition of commonSetBits API.
* int commonSetBits(int num);
*/

class Solution {
int findNumber() {

}
}
```

Scala Solution:

```
/**
* Definition of commonSetBits API (defined in the parent class Problem)
* def commonSetBits(num: Int): Int = {}
*/

class Solution extends Problem {
def findNumber(): Int = {

}

}
```

Elixir Solution:

```
# Definition of commonSetBits API.
# common_set_bits = fn
# num :: integer -> integer
# end
# Note that due to the limitations of the language, common_set_bits is passed to you as an anonymous function.
# To call it, you should use the dot notation. e.g., common_set_bits.(x)

defmodule Solution do
@spec find_number(common_set_bits :: (integer -> integer)) :: integer
def find_number(common_set_bits) do

end
end
```

Erlang Solution:

```
%% Definition of commonSetBits API.
%% -spec common_set_bits(Num :: integer()) -> integer().
%% common_set_bits(Num) ->
%% .

-spec find_number() -> integer().
find_number() ->
.
```

Racket Solution:

```
;; Definition of commonSetBits API.
#|
(define/contract (common-set-bits num)
(-> exact-integer? exact-integer?)
)
|#

(define/contract (find-number)
(-> exact-integer?)

)
```