# Problem 374: Guess Number Higher or Lower

## Problem Information

**Difficulty:** Easy
**Acceptance Rate:** 0.00%
**Paid Only:** No

## Problem Description

We are playing the Guess Game. The game is as follows:

I pick a number from

1

to

n

. You have to guess which number I picked (the number I picked stays the same throughout the game).

Every time you guess wrong, I will tell you whether the number I picked is higher or lower than your guess.

You call a pre-defined API

int guess(int num)

, which returns three possible results:

-1

: Your guess is higher than the number I picked (i.e.

num > pick

).

1

: Your guess is lower than the number I picked (i.e.

num < pick

).

0

: your guess is equal to the number I picked (i.e.

num == pick

).

Return

the number that I picked

.

Example 1:

Input:

n = 10, pick = 6

Output:

6

Example 2:

Input:

n = 1, pick = 1

Output:

1

Example 3:

Input:

n = 2, pick = 1

Output:

1

Constraints:

1 <= n <= 2

31

- 1

1 <= pick <= n

## Code Snippets

**C++:**

```
/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/
```

```
class Solution {
public:
int guessNumber(int n) {

}
};
```

## Java:

```
/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

public class Solution extends GuessGame {
public int guessNumber(int n) {

}
}
```

## Python3:

```
# The guess API is already defined for you.
# @param num, your guess
# @return -1 if num is higher than the picked number
# 1 if num is lower than the picked number
# otherwise return 0
# def guess(num: int) -> int:

class Solution:
def guessNumber(self, n: int) -> int:
```

## Python:

```
# The guess API is already defined for you.
# @param num, your guess
# @return -1 if num is higher than the picked number
# 1 if num is lower than the picked number
# otherwise return 0
# def guess(num):

class Solution(object):
def guessNumber(self, n):
"""
:type n: int
:rtype: int
"""
```

### JavaScript:

```
/**
* Forward declaration of guess API.
* @param {number} num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* var guess = function(num) {}
*/

/**
* @param {number} n
* @return {number}
*/
var guessNumber = function(n) {

};
```

### TypeScript:

```
/**
* Forward declaration of guess API.
* @param {number} num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* var guess = function(num) {}
*/
```

```
function guessNumber(n: number): number {

};
```

**C#:**

```
/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

public class Solution : GuessGame {
public int GuessNumber(int n) {

}
}
```

**C:**

```
/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

int guessNumber(int n){

}
```

**Go:**

```
/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* func guess(num int) int;
*/

func guessNumber(n int) int {

}
```

### Kotlin:

```
/**
* The API guess is defined in the parent class.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* fun guess(num:Int):Int {}
*/

class Solution:GuessGame() {
override fun guessNumber(n:Int):Int {

}
}
```

### Swift:

```
/**
* Forward declaration of guess API.
* @param num -> your guess number
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* func guess(_ num: Int) -> Int
*/

class Solution : GuessGame {
func guessNumber(_ n: Int) -> Int {

}
}
```

### Rust:

```
/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* unsafe fn guess(num: i32) -> i32 {}
*/

impl Solution {
unsafe fn guessNumber(n: i32) -> i32 {

}
}
```

### Ruby:

```
# The guess API is already defined for you.
# @param num, your guess
# @return -1 if num is higher than the picked number
# 1 if num is lower than the picked number
# otherwise return 0
# def guess(num)

def guessNumber(n)

end
```

### PHP:

```
/**
* The API guess is defined in the parent class.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* public function guess($num){}
*/

class Solution extends GuessGame {
/**
* @param Integer $n
* @return Integer
*/
function guessNumber($n) {

}
}
```

### Scala:

```
/**
* The API guess is defined in the parent class.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* def guess(num: Int): Int = {}
*/

class Solution extends GuessGame {
def guessNumber(n: Int): Int = {

}
}
```

## Solutions

### C++ Solution:

```
/*
* Problem: Guess Number Higher or Lower
* Difficulty: Easy
* Tags: search
*
* 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
*/

/**
```

```
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

class Solution {
public:
int guessNumber(int n) {

}
};
```

### Java Solution:

```
/**
* Problem: Guess Number Higher or Lower
* Difficulty: Easy
* Tags: search
*
* 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
*/

/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

public class Solution extends GuessGame {
public int guessNumber(int n) {

}
}
```

### Python3 Solution:

```
"""
Problem: Guess Number Higher or Lower
Difficulty: Easy
Tags: search

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
"""

# The guess API is already defined for you.
# @param num, your guess
# @return -1 if num is higher than the picked number
# 1 if num is lower than the picked number
# otherwise return 0
# def guess(num: int) -> int:

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

### Python Solution:

```
# The guess API is already defined for you.
# @param num, your guess
# @return -1 if num is higher than the picked number
# 1 if num is lower than the picked number
# otherwise return 0
# def guess(num):

class Solution(object):
def guessNumber(self, n):
"""
:type n: int
:rtype: int
"""
```

### JavaScript Solution:

```
/**
* Problem: Guess Number Higher or Lower
* Difficulty: Easy
* Tags: search
*
* 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
*/

/**
* Forward declaration of guess API.
* @param {number} num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* var guess = function(num) {}
*/

/**
* @param {number} n
* @return {number}
*/
var guessNumber = function(n) {

};
```

### TypeScript Solution:

```
/**
* Problem: Guess Number Higher or Lower
* Difficulty: Easy
* Tags: search
*
* 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
*/

/**
* Forward declaration of guess API.
* @param {number} num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* var guess = function(num) {}
*/


function guessNumber(n: number): number {

};
```

## C# Solution:

```
/*
* Problem: Guess Number Higher or Lower
* Difficulty: Easy
* Tags: search
*
* 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
*/

/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

public class Solution : GuessGame {
public int GuessNumber(int n) {

}
}
```

## C Solution:

```
/*
* Problem: Guess Number Higher or Lower
* Difficulty: Easy
* Tags: search
*
* 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
*/

/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* int guess(int num);
*/

int guessNumber(int n){

}
```

### Go Solution:

```
// Problem: Guess Number Higher or Lower
// Difficulty: Easy
// Tags: search
//
// 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

/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* func guess(num int) int;
*/
```

```
func guessNumber(n int) int {

}
```

### Kotlin Solution:

```
/**
* The API guess is defined in the parent class.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* fun guess(num:Int):Int {}
*/

class Solution:GuessGame() {
override fun guessNumber(n:Int):Int {

}
}
```

### Swift Solution:

```
/**
* Forward declaration of guess API.
* @param num -> your guess number
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* func guess(_ num: Int) -> Int
*/

class Solution : GuessGame {
func guessNumber(_ n: Int) -> Int {

}
}
```

### Rust Solution:

```
// Problem: Guess Number Higher or Lower
// Difficulty: Easy
// Tags: search
//
// 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

/**
* Forward declaration of guess API.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* unsafe fn guess(num: i32) -> i32 {}
*/

impl Solution {
unsafe fn guessNumber(n: i32) -> i32 {

}
}
```

### Ruby Solution:

```
# The guess API is already defined for you.
# @param num, your guess
# @return -1 if num is higher than the picked number
# 1 if num is lower than the picked number
# otherwise return 0
# def guess(num)

def guessNumber(n)

end
```

### PHP Solution:

```
/**
* The API guess is defined in the parent class.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* public function guess($num){}
*/

class Solution extends GuessGame {
/**
* @param Integer $n
* @return Integer
*/
function guessNumber($n) {

}
}
```

### Scala Solution:

```
/**
* The API guess is defined in the parent class.
* @param num your guess
* @return -1 if num is higher than the picked number
* 1 if num is lower than the picked number
* otherwise return 0
* def guess(num: Int): Int = {}
*/

class Solution extends GuessGame {
def guessNumber(n: Int): Int = {

}
}
```