Problem 1523: Count Odd Numbers in an Interval Range

Problem Information

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

Problem Description

Given two non-negative integers

low

and

high

. Return the

count of odd numbers between

low

and

high

(inclusive)

.

Example 1:

Input:

low = 3, high = 7

Output:

3

Explanation:

The odd numbers between 3 and 7 are [3,5,7].

Example 2:

Input:

low = 8, high = 10

Output:

1

Explanation:

The odd numbers between 8 and 10 are [9].

Constraints:

0 <= low <= high <= 10^9

Code Snippets

C++:

```
class Solution {
public:
int countOdds(int low, int high) {

}
};
```

Java:

```
class Solution {
public int countOdds(int low, int high) {

}
}
```

Python3:

```
class Solution:
def countOdds(self, low: int, high: int) -> int:
```

Python:

```
class Solution(object):
def countOdds(self, low, high):
"""
:type low: int
:type high: int
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number} low
* @param {number} high
* @return {number}
*/
var countOdds = function(low, high) {

};
```

TypeScript:

```
function countOdds(low: number, high: number): number {

};
```

C#:

```
public class Solution {
public int CountOdds(int low, int high) {

}
}
```

C:

```


int countOdds(int low, int high){

}
```

Go:

```
func countOdds(low int, high int) int {

}
```

Kotlin:

```
class Solution {
fun countOdds(low: Int, high: Int): Int {

}
}
```

Swift:

```
class Solution {
func countOdds(_ low: Int, _ high: Int) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn count_odds(low: i32, high: i32) -> i32 {

}
}
```

Ruby:

```
# @param {Integer} low
# @param {Integer} high
# @return {Integer}
def count_odds(low, high)

end
```

PHP:

```
class Solution {

/**
* @param Integer $low
* @param Integer $high
* @return Integer
*/
function countOdds($low, $high) {

}
}
```

Scala:

```
object Solution {
def countOdds(low: Int, high: Int): Int = {

}
}
```

Solutions

C++ Solution:

```
/*
* Problem: Count Odd Numbers in an Interval Range
* Difficulty: Easy
* Tags: math
*
* 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
*/

class Solution {
public:
int countOdds(int low, int high) {

}
};
```

Java Solution:

```
/**
* Problem: Count Odd Numbers in an Interval Range
* Difficulty: Easy
* Tags: math
*
* 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
*/

class Solution {
public int countOdds(int low, int high) {

}
}
```

Python3 Solution:

```
"""
Problem: Count Odd Numbers in an Interval Range
Difficulty: Easy
Tags: math

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
"""
```

```
class Solution:
def countOdds(self, low: int, high: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def countOdds(self, low, high):
"""
:type low: int
:type high: int
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Count Odd Numbers in an Interval Range
* Difficulty: Easy
* Tags: math
*
* 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
*/

/**
* @param {number} low
* @param {number} high
* @return {number}
*/
var countOdds = function(low, high) {

};
```

TypeScript Solution:

```
/**
* Problem: Count Odd Numbers in an Interval Range
* Difficulty: Easy
```

```
* Tags: math
*
* 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
*/

function countOdds(low: number, high: number): number {

};
```

C# Solution:

```
/*
* Problem: Count Odd Numbers in an Interval Range
* Difficulty: Easy
* Tags: math
*
* 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
*/

public class Solution {
public int CountOdds(int low, int high) {

}
}
```

C Solution:

```
/*
* Problem: Count Odd Numbers in an Interval Range
* Difficulty: Easy
* Tags: math
*
* 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
*/
```

```
int countOdds(int low, int high){

}
```

Go Solution:

```
// Problem: Count Odd Numbers in an Interval Range
// Difficulty: Easy
// Tags: math
//
// 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

func countOdds(low int, high int) int {

}
```

Kotlin Solution:

```
class Solution {
fun countOdds(low: Int, high: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func countOdds(_ low: Int, _ high: Int) -> Int {

}
}
```

Rust Solution:

```
// Problem: Count Odd Numbers in an Interval Range
// Difficulty: Easy
// Tags: math
```

```
//
// 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

impl Solution {
pub fn count_odds(low: i32, high: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer} low
# @param {Integer} high
# @return {Integer}
def count_odds(low, high)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer $low
* @param Integer $high
* @return Integer
*/
function countOdds($low, $high) {

}
}
```

Scala Solution:

```
object Solution {
def countOdds(low: Int, high: Int): Int = {

}
}
```