Find position of the only set bit - GeeksforGeeks

Source: https://www.geeksforgeeks.org/find-position-of-the-only-set-bit/

Courses Tutorials Practice Jobs DSA Tutorial Interview Questions Quizzes Must Do Advanced DSA System Design Aptitude Puzzles Interview Corner DSA Python Technical Scripter 2026 Explore DSA Fundamentals Logic Building Problems Analysis of Algorithms Data Structures Array Data Structure String in Data Structure Hashing in Data Structure Linked List Data Structure Stack Data Structure Queue Data Structure Tree Data Structure Graph Data Structure Trie Data Structure Algorithms Searching Algorithms Sorting Algorithms Introduction to Recursion Greedy Algorithms Tutorial Graph Algorithms Dynamic Programming or DP Bitwise Algorithms Advanced Segment Tree Binary Indexed Tree or Fenwick Tree Square Root (Sqrt) Decomposition Algorithm Binary Lifting Geometry Interview Preparation Interview Corner GfG160 Practice Problem GeeksforGeeks Practice - Leading Online Coding Platform Problem of The Day - Develop the Habit of Coding DSA Course 90% Refund Find position of the only set bit Last Updated : 23 Jul, 2025 Given a number n containing only 1 set bit in its binary representation, the task is to find the position of the only set bit. If there are 0 or more than 1 set bits, then return -1. Note: Position of set bit '1' should be counted starting with 1 from the LSB side in the binary representation of the number. Examples:- Input : n = 2 Output : 2 Explanation : Binary representation of 2 is 10. We can observe that the only set bit is at position 2 from LSB. Input : n = 5 Output : -1 Explanation : Binary representation of 5 is 101. There are 2 set bits, so return -1. Try it on GfG Practice Condition for numbers having only 1 set bit: Numbers having only one set bit will be a power of 2 number (example, 2 0 = 1, 2 1 = 10, 2 2 = 100, 2 3 = 1000). When you subtract 1 from such a number, all bits after the set bit (including the set bit itself) flip (e.g., 4 = 100 , 3 = 011). Performing a bitwise AND between n and n-1 results in 0 if n is a power of 2, as the single set bit cancels out. Refer to Program to find whether a given number is power of 2 for various approaches. Using Left Shift Operator - O(log(n)) time and O(1) space The idea is to use a loop where we left shift the number 1 and perform a bitwise AND operation with n . If the result is non-zero, the position of the set bit is determined by the number of shifts performed. C++ // C++ program to Find position // of the only set bit #include <bits/stdc++.h> using namespace std ; // Function to find set bit // Using left shift operator int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return -1 ; int pos = 1 ; int val = 1 ; while ((val & n) == 0) { val = val << 1 ; pos ++ ; } return pos ; } int main () { int n = 2 ; cout << findPosition (n); return 0 ; } Java // Java program to Find position // of the only set bit class GfG { // Function to find set bit // Using left shift operator static int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return - 1 ; int pos = 1 ; int val = 1 ; while ((val & n) == 0) { val = val << 1 ; pos ++ ; } return pos ; } public static void main (String [] args) { int n = 2 ; System . out . println (findPosition (n)); } } Python # Python program to Find position # of the only set bit # Function to find set bit # Using left shift operator def findPosition (n): # Check if n has exactly one set bit if n == 0 or (n & (n - 1)) != 0 : return - 1 pos = 1 val = 1 while (val & n) == 0 : val = val << 1 pos += 1 return pos if __name__ == "__main__" : n = 2 print (findPosition (n)) C# // C# program to Find position // of the only set bit using System ; class GfG { // Function to find set bit // Using left shift operator static int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return - 1 ; int pos = 1 ; int val = 1 ; while ((val & n) == 0) { val = val << 1 ; pos ++ ; } return pos ; } static void Main () { int n = 2 ; Console . WriteLine (findPosition (n)); } } JavaScript // JavaScript program to Find position // of the only set bit // Function to find set bit // Using left shift operator function findPosition (n) { // Check if n has exactly one set bit if (n === 0 || (n & (n - 1)) !== 0) return - 1 ; let pos = 1 ; let val = 1 ; while ((val & n) === 0) { val = val << 1 ; pos ++ ; } return pos ; } let n = 2 ; console . log (findPosition (n)); Output 2 Using Right Shift Operator - O(log(n)) time and O(1) space The idea is to right shift the number n until the rightmost bit becomes 1 . The number of shifts required to reach this point gives the position of the set bit. C++ // C++ program to Find position // of the only set bit #include <bits/stdc++.h> using namespace std ; // Function to find set bit // using right shift operator. int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return -1 ; int pos = 1 ; while ((n & 1) == 0) { n = n >> 1 ; pos ++ ; } return pos ; } int main () { int n = 2 ; cout << findPosition (n); return 0 ; } Java // Java program to Find position // of the only set bit class GfG { // Function to find set bit // using right shift operator. static int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return - 1 ; int pos = 1 ; while ((n & 1) == 0) { n = n

>> 1 ; pos ++ ; } return pos ; } public static void main (String [] args) { int n = 2 ; System . out . println (findPosition (n)); } } Python # Python program to Find position # of the only set bit # Function to find set bit # using right shift operator. def findPosition (n): # Check if n has exactly one set bit if n == 0 or (n & (n - 1)) != 0 : return - 1 pos = 1 while (n & 1) == 0 : n = n >> 1 pos += 1 return pos if __name__ == "__main__" : n = 2 print (findPosition (n)) C# // C# program to Find position // of the only set bit using System ; class GfG { // Function to find set bit // using right shift operator. static int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return - 1 ; int pos = 1 ; while ((n & 1) == 0) { n = n >> 1 ; pos ++ ; } return pos ; } static void Main () { int n = 2 ; Console . WriteLine (findPosition (n)); } } JavaScript // JavaScript program to Find position // of the only set bit // Function to find set bit // using right shift operator. function findPosition (n) { // Check if n has exactly one set bit if (n === 0 || (n & (n - 1)) !== 0) return - 1 ; let pos = 1 ; while ((n & 1) === 0) { n = n >> 1 ; pos ++ ; } return pos ; } let n = 2 ; console . log (findPosition (n)); Output 2 Using Log Operator - O(log n) time and O(1) space The idea is to use the mathematical property that the position of the only set bit in a number n (which is a power of 2) can be found by taking the base-2 logarithm of n and adding 1 (since the position is 1-based). C++ // C++ program to Find position // of the only set bit #include <bits/stdc++.h> using namespace std ; // Function to find set bit // using log operator. int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return -1 ; return log2 (n) + 1 ; } int main () { int n = 2 ; cout << findPosition (n); return 0 ; } Java // Java program to Find position // of the only set bit class GfG { // Function to find set bit // using log operator. static int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return - 1 ; return (int)(Math . log (n) / Math . log (2)) + 1 ; } public static void main (String [] args) { int n = 2 ; System . out . println (findPosition (n)); } } Python # Python program to Find position # of the only set bit import math # Function to find set bit # using log operator. def findPosition (n): # Check if n has exactly one set bit if n == 0 or (n & (n - 1)) != 0 : return - 1 return int (math . log2 (n)) + 1 if __name__ == "__main__" : n = 2 print (int (findPosition (n))) C# // C# program to Find position // of the only set bit using System ; class GfG { // Function to find set bit // using log operator. static int findPosition (int n) { // Check if n has exactly one set bit if (n == 0 || (n & (n - 1)) != 0) return - 1 ; return (int)(Math . Log (n) / Math . Log (2)) + 1 ; } static void Main () { int n = 2 ; Console . WriteLine (findPosition (n)); } } JavaScript // JavaScript program to Find position // of the only set bit // Function to find set bit // using log operator. function findPosition (n) { // Check if n has exactly one set bit if (n === 0 || (n & (n - 1)) !== 0) return - 1 ; return Math . log2 (n) + 1 ; } let n = 2 ; console . log (Math . floor (findPosition (n))); Output 2 Comment Article Tags: Article Tags: Bit Magic DSA Microsoft