Introduction to Primality Test and School Method - GeeksforGeeks

Source: https://www.geeksforgeeks.org/primality-test-set-1-introduction-and-school-method/

Introduction to Primality Test and School Method Last Updated : 13 Feb, 2025 Given a positive integer, check if the number is prime or not. A prime is a natural number greater than 1 that has no positive divisors other than 1 and itself. Examples of the first few prime numbers are {2, 3, 5, ...} Examples : Input: n = 11 Output: true Input: n = 15 Output: false Input: n = 1 Output: false Try it on GfG Practice School Method A simple solution is to iterate through all numbers from 2 to n-1 and for every number check if it divides n. If we find any number that divides, we return false. C++ // A school method based C++ program to check if a // number is prime #include <bits/stdc++.h> using namespace std ; bool isPrime (int n) { // Corner case if (n <= 1) return false ; // Check from 2 to n-1 for (int i = 2 ; i < n ; i ++) if (n % i == 0) return false ; return true ; } // Driver Program to test above function int main () { isPrime (11) ? cout << " true \n " : cout << " false \n " ; isPrime (15) ? cout << " true \n " : cout << " false \n " ; return 0 ; } Java // A school method based JAVA program // to check if a number is prime class GFG { static boolean isPrime (int n) { // Corner case if (n <= 1) return false ; // Check from 2 to n-1 for (int i = 2 ; i < n ; i ++) if (n % i == 0) return false ; return true ; } // Driver Program public static void main (String args []) { if (isPrime (11)) System . out . println (" true"); else System . out . println (" false"); if (isPrime (15)) System . out . println (" true"); else System . out . println (" false"); } } Python # A school method based Python3 # program to check if a number # is prime def isPrime (n): # Corner case if n <= 1 : return False # Check from 2 to n-1 for i in range (2 , n): if n % i == 0 : return False return True # Driver Program to test above function print ("true") if isPrime (11) else print ("false") print ("true") if isPrime (14) else print ("false") C# // A optimized school method based C# // program to check if a number is prime using System ; namespace prime { public class GFG { public static bool isprime (int n) { // Corner cases if (n <= 1) return false ; for (int i = 2 ; i < n ; i ++) if (n % i == 0) return false ; return true ; } // Driver program public static void Main () { if (isprime (11)) Console . WriteLine ("true"); else Console . WriteLine ("false"); if (isprime (15)) Console . WriteLine ("true"); else Console . WriteLine ("false"); } } } JavaScript < script > // A school method based Javascript program to check if a // number is prime function isPrime (n) { // Corner case if (n <= 1) return false ; // Check from 2 to n-1 for (let i = 2 ; i < n ; i ++) if (n % i == 0) return false ; return true ; } // Driver Program to test above function isPrime (11) ? document . write (" true" + "<br>") : document . write (" false" + "<br>"); isPrime (15) ? document . write (" true" + "<br>") : document . write (" false" + "<br>"); < /script> PHP <?php // A school method based PHP // program to check if a number // is prime function isPrime ($n) { // Corner case if ($n <= 1) return false ; // Check from 2 to n-1 for ($i = 2 ; $i < $n ; $i ++) if ($n % $i == 0) return false ; return true ; } // Driver Code $tet = isPrime (11) ? " true \n " : " false \n " ; echo $tet ; $tet = isPrime (15) ? " true \n " : " false \n " ; echo $tet ; ?> Output true false Time complexity: O(n) Auxiliary Space: O(1) Optimized School Method We can do the following optimizations: Instead of checking till n, we can check till √n because a larger factor of n must be a multiple of a smaller factor that has been already checked. The implementation of this method is as follows: C++ // Optimised school method based C++ program to check if a // number is prime #include <bits/stdc++.h> using namespace std ; bool isPrime (int n) { // Corner case if (n <= 1) return false ; // Check from 2 to square root of n for (int i = 2 ; i <= sqrt (n); i ++) if (n % i == 0) return false ; return true ; } // Driver Program to test above function int main () { isPrime (11) ? cout << " true \n " : cout << "

false \n " ; isPrime (15) ? cout << " true \n " : cout << " false \n " ; return 0 ; } // This code is contributed by Vikash Sangai Java // Optimised school method based JAVA program // to check if a number is prime class GFG { static boolean isPrime (int n) { // Corner case if (n <= 1) return false ; // Check from 2 to square root of n for (int i = 2 ; i * i <= n ; i ++) if (n % i == 0) return false ; return true ; } // Driver Program public static void main (String args []) { if (isPrime (11)) System . out . println (" true"); else System . out . println (" false"); if (isPrime (15)) System . out . println (" true"); else System . out . println (" false"); } } // This code is contributed by Vikash Sangai Python # Optimised school method based PYTHON program # to check if a number is prime # import the math module import math # function to check whether the number is prime or not def isPrime (n): # Corner case if (n <= 1): return False # Check from 2 to square root of n for i in range (2 , int (math . sqrt (n)) + 1): if (n % i == 0): return False return True # Driver Program to test above function print ("true") if isPrime (11) else print ("false") print ("true") if isPrime (15) else print ("false") # This code is contributed by bhoomikavemula C# // Optimised school method based C# // program to check if a number is prime using System ; namespace prime { public class GFG { public static bool isprime (int n) { // Corner cases if (n <= 1) return false ; for (int i = 2 ; i * i <= n ; i ++) if (n % i == 0) return false ; return true ; } // Driver program public static void Main () { if (isprime (11)) Console . WriteLine ("true"); else Console . WriteLine ("false"); if (isprime (15)) Console . WriteLine ("true"); else Console . WriteLine ("false"); } } } // This code is contributed by Vikash Sangai JavaScript < script > // JavaScript code for the above approach function isPrime (n) { // Corner case if (n <= 1) return false ; // Check from 2 to square root of n for (let i = 2 ; i * i <= n ; i ++) if (n % i == 0) return false ; return true ; } // Driver Code if (isPrime (11)) document . write (" true" + "<br/>"); else document . write (" false" + "<br/>"); if (isPrime (15)) document . write (" true" + "<br/>"); else document . write (" false" + "<br/>"); // This code is contributed by sanjoy_62. < /script> Output true false Time Complexity: O(√n) Auxiliary Space: O(1) Another approach It is based on the fact that all primes greater than 3 are of the form 6k ± 1, where k is any integer greater than 0. This is because all integers can be expressed as (6k + i), where i = −1, 0, 1, 2, 3, or 4. And note that 2 divides (6k + 0), (6k + 2), and (6k + 4) and 3 divides (6k + 3). So, a more efficient method is to test whether n is divisible by 2 or 3, then to check through all numbers of the form 6k ± 1 <= √n. This is 3 times faster than testing all numbers up to √n. C++ // C++ program to check the given number // is prime or not #include <bits/stdc++.h> using namespace std ; // Function to check if the given number // is prime or not. bool isPrime (int n) { if (n == 2 || n == 3) return true ; if (n <= 1 || n % 2 == 0 || n % 3 == 0) return false ; // To check through all numbers of the form 6k ± 1 for (int i = 5 ; i * i <= n ; i += 6) { if (n % i == 0 || n % (i + 2) == 0) return false ; } return true ; } // Driver Code int main () { isPrime (11) ? cout << " true \n " : cout << " false \n " ; isPrime (15) ? cout << " true \n " : cout << " false \n " ; return 0 ; } Java // JAVA program to check the given number // is prime or not class GFG { static boolean isPrime (int n) { // since 2 and 3 are prime if (n == 2 || n == 3) return true ; // if n<=1 or divided by 2 or 3 then it can not be prime if (n <= 1 || n % 2 == 0 || n % 3 == 0) return false ; // To check through all numbers of the form 6k ± 1 for (int i = 5 ; i * i <= n ; i += 6) { if (n % i == 0 || n % (i + 2) == 0) return false ; } return true ; } // Driver Program public static void main (String args []) { if (isPrime (11)) System . out . println (" true"); else System . out . println (" false"); if (isPrime (15)) System . out . println (" true"); else System . out . println (" false"); } } // This code is contributed by Ujjwal Kumar Bhardwaj Python # Python program to check the given number # is prime or not # Function to check if the given number # is prime or not. import math def isPrime (n): if n == 2 or n == 3 : return True elif n <= 1 or n % 2 == 0 or n % 3 == 0 : return False # To check through all numbers of the form 6k ± 1 # until i <= square root of n, with step value 6 for i in range (5 , int (math . sqrt (n)) + 1 , 6): if n % i == 0 or n % (i + 2) == 0 : return False return True # # Driver code print (isPrime (11)) print (isPrime (20)) # # This code is contributed by Harsh Master C# // C# program to check the given number // is prime or not using System ; class GFG { static bool isPrime (int n) { // since 2 and 3 are prime if (n == 2 || n == 3) return true ; // if n<=1 or divided by 2 or 3 then it can not be prime if (n <= 1 || n % 2 == 0 || n % 3 == 0) return false ; // To check through all numbers of the form 6k ± 1 for (int i = 5 ; i * i <= n ; i += 6) { if (n % i == 0 || n % (i + 2) == 0) return false ; } return true ; } // Driver Program public static void Main (String [] args) { if (isPrime (11)) Console . WriteLine (" true"); else Console . WriteLine (" false"); if (isPrime (15)) Console . WriteLine (" true"); else Console . WriteLine (" false"); } } // This code is contributed by Aman Kumar JavaScript < script > // JavaScript program to check the given number // is prime or not // Function to check if the given number // is prime or not. function isPrime (n) { if (n == 2 || n == 3) return true ; if (n <= 1 || n % 2 == 0 || n % 3 == 0) return false ; // To check through all numbers of the form 6k ± 1 for (let i = 5 ; i * i <= n ; i += 6) { if (n % i == 0 || n % (i + 2) == 0) return false ; } return true ; } // Driver Code isPrime (11) ? document . write (" true" + "<br/>"

) : document . write (" false" + "<br/>"); isPrime (15) ? document . write (" true" + "<br/>") : document . write (" false" + "<br/>"); < /script> Output true false Time Complexity: O(√n) Auxiliary Space: O(1) Comment Article Tags: Article Tags: Mathematical DSA Prime Number number-theory