Search, Insert, and Delete in an Unsorted Array | Array Operations - GeeksforGeeks

Source: https://www.geeksforgeeks.org/search-insert-and-delete-in-an-unsorted-array/

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 Search, Insert, and Delete in an Unsorted Array | Array Operations Last Updated : 23 Jul, 2025 In this post, a program to search, insert, and delete operations in an unsorted array is discussed. Search Operation: In an unsorted array, the search operation can be performed by linear traversal from the first element to the last element. Coding implementation of the search operation: Try it on GfG Practice C++ // C++ program to implement linear // search in unsorted array #include <bits/stdc++.h> using namespace std ; // Function to implement search operation int findElement (int arr [], int n , int key) { int i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; // If the key is not found return -1 ; } // Driver's Code int main () { int arr [] = { 12 , 34 , 10 , 6 , 40 }; int n = sizeof (arr) / sizeof (arr [0]); // Using a last element as search element int key = 40 ; // Function call int position = findElement (arr , n , key); if (position == -1) cout << "Element not found" ; else cout << "Element Found at Position: " << position + 1 ; return 0 ; } // This code is contributed // by Akanksha Rai C // C program to implement linear // search in unsorted array #include <stdio.h> // Function to implement search operation int findElement (int arr [], int n , int key) { int i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; // If the key is not found return -1 ; } // Driver's Code int main () { int arr [] = { 12 , 34 , 10 , 6 , 40 }; int n = sizeof (arr) / sizeof (arr [0]); // Using a last element as search element int key = 40 ; // Function call int position = findElement (arr , n , key); if (position == -1) printf ("Element not found"); else printf ("Element Found at Position: %d" , position + 1); return 0 ; } Java // Java program to implement linear // search in unsorted arrays class Main { // Function to implement // search operation static int findElement (int arr [] , int n , int key) { for (int i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; // If the key is not found return - 1 ; } // Driver's Code public static void main (String args []) { int arr [] = { 12 , 34 , 10 , 6 , 40 }; int n = arr . length ; // Using a last element as search element int key = 40 ; // Function call int position = findElement (arr , n , key); if (position == - 1) System . out . println ("Element not found"); else System . out . println ("Element Found at Position: " + (position + 1)); } } Python # Python program for searching in # unsorted array def findElement (arr , n , key): for i in range (n): if (arr [i] == key): return i # If the key is not found return - 1 # Driver's code if __name__ == '__main__' : arr = [12 , 34 , 10 , 6 , 40] key = 40 n = len (arr) # search operation index = findElement (arr , n , key) if index != - 1 : print ("Element Found at position: " + str (index + 1)) else : print ("Element not found") # Thanks to Aditi Sharma for contributing # this code C# // C# program to implement linear // search in unsorted arrays using System ; class main { // Function to implement // search operation static int findElement (int [] arr , int n , int key) { for (int i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; // If the key is not found return - 1 ; } // Driver Code public static void Main () { int [] arr = { 12 , 34 , 10 , 6 , 40 }; int n = arr . Length ; // Using a last element as // search element int key = 40 ; int position = findElement (arr , n , key); if (position == - 1) Console . WriteLine ("Element not found"); else Console . WriteLine ("Element Found at Position: " + (position + 1)); } } // This code is contributed by vt_m. JavaScript // Javascript program to implement linear // search in unsorted array // Function to implement search operation function findElement (arr , n , key) { let i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; return - 1 ; } // Driver program let arr = [12 , 34 , 10 , 6 , 40]; let n = arr . length ; // Using a last element as search element let key = 40 ; let position = findElement (arr , n , key); if (position == - 1) console . log ("Element not found"); else console . log ("Element Found at Position: " + (position + 1)); PHP <?php // PHP program to implement linear //

search in unsorted array // Function to implement // search operation function findElement ($arr , $n , $key) { $i ; for ($i = 0 ; $i < $n ; $i ++) if ($arr [$i] == $key) return $i ; // If the key is not found return - 1 ; } // Driver Code $arr = array (12 , 34 , 10 , 6 , 40); $n = sizeof ($arr); // Using a last element // as search element $key = 40 ; $position = findElement ($arr , $n , $key); if ($position == - 1) echo ("Element not found"); else echo ("Element Found at Position: " . ($position + 1)); // This code is contributed by Ajit. ?> Output Element Found at Position: 5 Time Complexity: O(N) Auxiliary Space: O(1) Insert Operation: 1. Insert at the end: In an unsorted array, the insert operation is faster as compared to a sorted array because we don't have to care about the position at which the element is to be placed. Coding implementation of inserting an element at the end: C++ #include <iostream> using namespace std ; // Inserts a key in arr[] of the given capacity. // n is the current size of arr[]. This // function returns n + 1 if insertion // is successful, else n. int insertEnd (int arr [], int n , int key , int capacity) { // Cannot insert more elements if n is // already more than or equal to capacity if (n >= capacity) return n ; arr [n] = key ; return (n + 1); } int main () { int arr [20] = { 12 , 16 , 20 , 40 , 50 , 70 }; int capacity = sizeof (arr) / sizeof (arr [0]); int n = 6 ; int i , key = 26 ; cout << "Before Insertion: " ; for (i = 0 ; i < n ; i ++) cout << arr [i] << " " ; // Inserting key n = insertEnd (arr , n , key , capacity); cout << " \n After Insertion: " ; for (i = 0 ; i < n ; i ++) cout << arr [i] << " " ; return 0 ; } C // C program to implement insert // operation in an unsorted array. #include <stdio.h> // Inserts a key in arr[] of given capacity. // n is current size of arr[]. This // function returns n + 1 if insertion // is successful, else n. int insertEnd (int arr [], int n , int key , int capacity) { // Cannot insert more elements if n is // already more than or equal to capacity if (n >= capacity) return n ; arr [n] = key ; return (n + 1); } // Driver Code int main () { int arr [20] = { 12 , 16 , 20 , 40 , 50 , 70 }; int capacity = sizeof (arr) / sizeof (arr [0]); int n = 6 ; int i , key = 26 ; printf (" \n Before Insertion: "); for (i = 0 ; i < n ; i ++) printf ("%d " , arr [i]); // Inserting key n = insertEnd (arr , n , key , capacity); printf (" \n After Insertion: "); for (i = 0 ; i < n ; i ++) printf ("%d " , arr [i]); return 0 ; } Java // Java program to implement insert // operation in an unsorted array. class Main { // Function to insert a given key in // the array. This function returns n+1 // if insertion is successful, else n. static int insertEnd (int arr [] , int n , int key , int capacity) { // Cannot insert more elements if n // is already more than or equal to // capacity if (n >= capacity) return n ; arr [n] = key ; return (n + 1); } // Driver Code public static void main (String [] args) { int [] arr = new int [20] ; arr [0] = 12 ; arr [1] = 16 ; arr [2] = 20 ; arr [3] = 40 ; arr [4] = 50 ; arr [5] = 70 ; int capacity = 20 ; int n = 6 ; int i , key = 26 ; System . out . print ("Before Insertion: "); for (i = 0 ; i < n ; i ++) System . out . print (arr [i] + " "); // Inserting key n = insertEnd (arr , n , key , capacity); System . out . print ("\n After Insertion: "); for (i = 0 ; i < n ; i ++) System . out . print (arr [i] + " "); } } Python # Python program for inserting # an element in an unsorted array # method to insert element def insertEnd (arr , element): arr . append (element) # Driver's code if __name__ == '__main__' : # declaring array and key to insert arr = [12 , 16 , 20 , 40 , 50 , 70] key = 26 # array before inserting an element print ("Before Inserting: ") print (arr) # array after Inserting element insertEnd (arr , key) print ("After Inserting: ") print (arr) # Thanks to Aditi Sharma for contributing # this code C# // C# program to implement insert // operation in an unsorted array. using System ; class main { // Function to insert a given // key in the array. This // function returns n + 1 // if insertion is successful, // else n. static int insertEnd (int [] arr , int n , int key , int capacity) { // Cannot insert more elements // if n is already more than // or equal to capacity if (n >= capacity) return n ; arr [n] = key ; return (n + 1); } // Driver Code public static void Main () { int [] arr = new int [20]; arr [0] = 12 ; arr [1] = 16 ; arr [2] = 20 ; arr [3] = 40 ; arr [4] = 50 ; arr [5] = 70 ; int capacity = 20 ; int n = 6 ; int i , key = 26 ; Console . Write ("Before Insertion: "); for (i = 0 ; i < n ; i ++) Console . Write (arr [i] + " "); Console . WriteLine (); // Inserting key n = insertEnd (arr , n , key , capacity); Console . Write ("After Insertion: "); for (i = 0 ; i < n ; i ++) Console . Write (arr [i] + " "); } } // This code is contributed by vt_m. JavaScript // Javascript program to implement insert // operation in an unsorted array. // Function to insert a given // key in the array. This // function returns n + 1 // if insertion is successful, // else n. function insertEnd (arr , n , key , capacity) { // Cannot insert more elements // if n is already more than // or equal to capacity if (n >= capacity) return n ; arr [n] = key ; return (n + 1); } let arr = new Array (20); arr [0] = 12 ; arr [1] = 16 ; arr [2] = 20 ; arr [3] = 40 ; arr [4] = 50 ; arr [5] = 70 ; let capacity = 20 ; let n = 6 ; let i , key = 26 ; console . log ("Before Insertion: "); for (i = 0 ; i < n ; i ++) console . log (arr [i] + " "); console . log ("</br>"); // Inserting key n = insertEnd (arr , n , key , capacity); console . log ("After Insertion: "); for (i = 0 ; i < n ; i ++) console . log (arr [i] + " "); PHP <?php // PHP program to implement insert // operation in an unsorted array. // Inserts a key in arr[] of given // capacity. n is current size of arr[]. // This function returns n + 1 if // insertion is successful, else n. function insertEnd (& $arr , $n , $key , $capacity) { // Cannot insert more elements if n is // already more than or equal to capacity if ($n >= $capacity) return $n ;

array_push ($arr , $key); return ($n + 1); } // Driver Code $arr = array (12 , 16 , 20 , 40 , 50 , 70); $capacity = 20 ; $n = 6 ; $key = 26 ; echo "Before Insertion: " ; for ($i = 0 ; $i < $n ; $i ++) echo $arr [$i] . " " ; // Inserting key $n = insertEnd ($arr , $n , $key , $capacity); echo " \n After Insertion: " ; for ($i = 0 ; $i < $n ; $i ++) echo $arr [$i] . " " ; // This code is contributed by // Rajput-Ji ?> Output Before Insertion: 12 16 20 40 50 70 After Insertion: 12 16 20 40 50 70 26 Time Complexity: O(1) Auxiliary Space: O(1) 2. Insert at any position Insert operation in an array at any position can be performed by shifting elements to the right, which are on the right side of the required position Coding implementation of inserting an element at any position: C++ // C++ Program to Insert an element // at a specific position in an Array #include <bits/stdc++.h> using namespace std ; // Function to insert element // at a specific position void insertElement (int arr [], int n , int x , int pos) { // shift elements to the right // which are on the right side of pos for (int i = n - 1 ; i >= pos ; i --) arr [i + 1] = arr [i]; arr [pos] = x ; } // Driver's code int main () { int arr [15] = { 2 , 4 , 1 , 8 , 5 }; int n = 5 ; cout << "Before insertion : " ; for (int i = 0 ; i < n ; i ++) cout << arr [i] << " " ; cout << endl ; int x = 10 , pos = 2 ; // Function call insertElement (arr , n , x , pos); n ++ ; cout << "After insertion : " ; for (int i = 0 ; i < n ; i ++) cout << arr [i] << " " ; return 0 ; } C // C Program to Insert an element // at a specific position in an Array #include <stdio.h> // Function to insert element // at a specific position void insertElement (int arr [], int n , int x , int pos) { // shift elements to the right // which are on the right side of pos for (int i = n - 1 ; i >= pos ; i --) arr [i + 1] = arr [i]; arr [pos] = x ; } // Driver's code int main () { int arr [15] = { 2 , 4 , 1 , 8 , 5 }; int n = 5 ; printf ("Before insertion : "); for (int i = 0 ; i < n ; i ++) printf ("%d " , arr [i]); printf (" \n "); int x = 10 , pos = 2 ; // Function call insertElement (arr , n , x , pos); n ++ ; printf ("After insertion : "); for (int i = 0 ; i < n ; i ++) printf ("%d " , arr [i]); return 0 ; } Java /*package whatever //do not write package name here */ import java.io.* ; // Java Program to Insert an element // at a specific position in an Array class GFG { static void insertElement (int arr [] , int n , int x , int pos) { // shift elements to the right // which are on the right side of pos for (int i = n - 1 ; i >= pos ; i --) arr [i + 1] = arr [i] ; arr [pos] = x ; } public static void main (String [] args) { int arr [] = new int [15] ; arr [0] = 2 ; arr [1] = 4 ; arr [2] = 1 ; arr [3] = 8 ; arr [4] = 5 ; int n = 5 ; int x = 10 , pos = 2 ; System . out . print ("Before Insertion: "); for (int i = 0 ; i < n ; i ++) System . out . print (arr [i] + " "); // Inserting key at specific position insertElement (arr , n , x , pos); n += 1 ; System . out . print ("\n\nAfter Insertion: "); for (int i = 0 ; i < n ; i ++) System . out . print (arr [i] + " "); } } // This code is contributed by syedsarfarazahammed Python # python Program to Insert an element # at a specific position in an Array def insertElement (arr , n , x , pos) : # shift elements to the right # which are on the right side of pos for i in range (n - 1 , pos - 1 , - 1) : arr [i + 1] = arr [i] arr [pos] = x # Driver's code if __name__ == '__main__' : # Declaring array and key to delete # here -1 is for empty space arr = [2 , 4 , 1 , 8 , 5 , - 1 , - 1 , - 1 , - 1 , - 1 , - 1 , - 1 , - 1 , - 1 , - 1] n = 5 print ("Before insertion : ") for i in range (0 , n) : print (arr [i], end = ' ') print (" \n ") x = 10 ; pos = 2 ; # Function call insertElement (arr , n , x , pos); n += 1 print ("After insertion : ") for i in range (0 , n) : print (arr [i], end = ' ') #This Code is contributed by aditya942003patil C# // C# program to implement insert // operation in an unsorted array. using System ; class main { static void insertElement (int [] arr , int n , int x , int pos) { // shift elements to the right // which are on the right side of pos for (int i = n - 1 ; i >= pos ; i --) arr [i + 1] = arr [i]; arr [pos] = x ; } public static void Main () { int [] arr = new int [20]; arr [0] = 2 ; arr [1] = 4 ; arr [2] = 1 ; arr [3] = 8 ; arr [4] = 5 ; int x = 10 ; int n = 5 ; int pos = 2 ; Console . Write ("Before Insertion: "); for (int i = 0 ; i < n ; i ++) Console . Write (arr [i] + " "); Console . WriteLine (); // Inserting key at specific position insertElement (arr , n , x , pos); n += 1 ; Console . Write ("\n\nAfter Insertion: "); for (int i = 0 ; i < n ; i ++) Console . Write (arr [i] + " "); } } // This code is contributed by sourabhdalal0001 JavaScript // javascript Program to Insert an element // at a specific position in an Array function insertElement (arr , n , x , pos) { // shift elements to the right // which are on the right side of pos var i = n - 1 ; for (i ; i >= pos ; i --) { arr [i + 1] = arr [i]; } arr [pos] = x ; } var arr = Array (15). fill (0); arr [0] = 2 ; arr [1] = 4 ; arr [2] = 1 ; arr [3] = 8 ; arr [4] = 5 ; var n = 5 ; var x = 10 ; var pos = 2 ; console . log ("Before Insertion: "); var i = 0 ; for (i ; i < n ; i ++) { console . log (arr [i] + " "); } // Inserting key at specific position insertElement (arr , n , x , pos); n += 1 ; console . log ("\n\nAfter Insertion: "); i = 0 ; for (i ; i < n ; i ++) { console . log (arr [i] + " "); } PHP <?php $array = [2 , 4 , 1 , 8 , 5]; $element = 10 ; function addElementAtPos ($element , $position , & $array) { // Shift elements to the right to make space for the new element for ($i = count ($array) - 1 ; $i >= $position ; $i --) { $array [$i + 1] = $array [$i]; } // Insert the new element at the specified position $array [$position] = $element ; } echo 'Before: ' . implode (' ' , $array) . " \n " ; addElementAtPos ($element , 2 , $array); echo 'After: ' . implode (' ' , $array) . " \n " ; ?> Output Before insertion : 2 4 1 8 5 After insertion : 2 4 10 1 8 5 Time complexity: O(N) Auxiliary Space: O(1) Delete Operation: In the delete operation, the element to be deleted is searched using the linear search

, and then the delete operation is performed followed by shifting the elements. C++ // C++ program to implement delete operation in a // unsorted array #include <iostream> using namespace std ; // To search a key to be deleted int findElement (int arr [], int n , int key); // Function to delete an element int deleteElement (int arr [], int n , int key) { // Find position of element to be deleted int pos = findElement (arr , n , key); if (pos == -1) { cout << "Element not found" ; return n ; } // Deleting element int i ; for (i = pos ; i < n - 1 ; i ++) arr [i] = arr [i + 1]; return n - 1 ; } // Function to implement search operation int findElement (int arr [], int n , int key) { int i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; return -1 ; } // Driver's code int main () { int i ; int arr [] = { 10 , 50 , 30 , 40 , 20 }; int n = sizeof (arr) / sizeof (arr [0]); int key = 30 ; cout << "Array before deletion \n " ; for (i = 0 ; i < n ; i ++) cout << arr [i] << " " ; // Function call n = deleteElement (arr , n , key); cout << " \n\n Array after deletion \n " ; for (i = 0 ; i < n ; i ++) cout << arr [i] << " " ; return 0 ; } // This code is contributed by shubhamsingh10 C // C program to implement delete operation in a // unsorted array #include <stdio.h> // To search a key to be deleted int findElement (int arr [], int n , int key); // Function to delete an element int deleteElement (int arr [], int n , int key) { // Find position of element to be deleted int pos = findElement (arr , n , key); if (pos == -1) { printf ("Element not found"); return n ; } // Deleting element int i ; for (i = pos ; i < n - 1 ; i ++) arr [i] = arr [i + 1]; return n - 1 ; } // Function to implement search operation int findElement (int arr [], int n , int key) { int i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; return -1 ; } // Driver's code int main () { int i ; int arr [] = { 10 , 50 , 30 , 40 , 20 }; int n = sizeof (arr) / sizeof (arr [0]); int key = 30 ; printf ("Array before deletion \n "); for (i = 0 ; i < n ; i ++) printf ("%d " , arr [i]); // Function call n = deleteElement (arr , n , key); printf (" \n Array after deletion \n "); for (i = 0 ; i < n ; i ++) printf ("%d " , arr [i]); return 0 ; } Java // Java program to implement delete // operation in an unsorted array class Main { // function to search a key to // be deleted static int findElement (int arr [] , int n , int key) { int i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; return - 1 ; } // Function to delete an element static int deleteElement (int arr [] , int n , int key) { // Find position of element to be // deleted int pos = findElement (arr , n , key); if (pos == - 1) { System . out . println ("Element not found"); return n ; } // Deleting element int i ; for (i = pos ; i < n - 1 ; i ++) arr [i] = arr [i + 1] ; return n - 1 ; } // Driver's Code public static void main (String args []) { int i ; int arr [] = { 10 , 50 , 30 , 40 , 20 }; int n = arr . length ; int key = 30 ; System . out . println ("Array before deletion"); for (i = 0 ; i < n ; i ++) System . out . print (arr [i] + " "); // Function call n = deleteElement (arr , n , key); System . out . println ("\n\nArray after deletion"); for (i = 0 ; i < n ; i ++) System . out . print (arr [i] + " "); } } Python # Python program to delete an element # from an unsorted array # Driver's code if __name__ == '__main__' : # Declaring array and key to delete arr = [10 , 50 , 30 , 40 , 20] key = 30 print ("Array before deletion:") print (arr) # deletes key if found in the array # otherwise shows error not in list arr . remove (key) print ("Array after deletion") print (arr) # This code is contributed by Aditi Sharma. C# // C# program to implement delete // operation in an unsorted array using System ; class main { // Function to search a // key to be deleted static int findElement (int [] arr , int n , int key) { int i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; return - 1 ; } // Function to delete an element static int deleteElement (int [] arr , int n , int key) { // Find position of element // to be deleted int pos = findElement (arr , n , key); if (pos == - 1) { Console . WriteLine ("Element not found"); return n ; } // Deleting element int i ; for (i = pos ; i < n - 1 ; i ++) arr [i] = arr [i + 1]; return n - 1 ; } // Driver Code public static void Main () { int i ; int [] arr = { 10 , 50 , 30 , 40 , 20 }; int n = arr . Length ; int key = 30 ; Console . Write ("Array before deletion "); for (i = 0 ; i < n ; i ++) Console . Write (arr [i] + " "); Console . WriteLine (); // Function call n = deleteElement (arr , n , key); Console . Write ("Array after deletion "); for (i = 0 ; i < n ; i ++) Console . Write (arr [i] + " "); } } // This code is contributed by vt_m. JavaScript // Java script program to implement delete // operation in an unsorted array // function to search a key to // be deleted function findElement (arr , n , key) { let i ; for (i = 0 ; i < n ; i ++) if (arr [i] == key) return i ; return - 1 ; } // Function to delete an element function deleteElement (arr , n , key) { // Find position of element to be // deleted let pos = findElement (arr , n , key); if (pos == - 1) { console . log ("Element not found"); return n ; } // Deleting element let i ; for (i = pos ; i < n - 1 ; i ++) arr [i] = arr [i + 1]; return n - 1 ; } // Driver Code let i ; let arr = [10 , 50 , 30 , 40 , 20]; let n = arr . length ; let key = 30 ; console . log ("Array before deletion
"); for (i = 0 ; i < n ; i ++) console . log (arr [i] + " "); n = deleteElement (arr , n , key); console . log ("

Array after deletion
"); for (i = 0 ; i < n ; i ++) console . log (arr [i] + " "); PHP <?php // PHP program to implement delete // operation in an unsorted array // To search a key to be deleted function findElement (& $arr , $n , $key) { for ($i = 0 ; $i < $n ; $i ++) if ($arr [$i] == $key) return $i ; return - 1 ; } // Function to delete an element function deleteElement (& $arr , $n , $key) { // Find position of element to // be deleted $pos = findElement ($arr , $n , $key); if ($pos == - 1) { echo "Element not found" ; return $n ; } // Deleting element for ($i = $pos ; $i < $n - 1 ; $i ++) $arr [$i] = $arr

[$i + 1]; return $n - 1 ; } // Driver code $arr = array (10 , 50 , 30 , 40 , 20); $n = count ($arr); $key = 30 ; echo "Array before deletion \n " ; for ($i = 0 ; $i < $n ; $i ++) echo $arr [$i] . " " ; // Function call $n = deleteElement ($arr , $n , $key); echo " \n Array after deletion \n " ; for ($i = 0 ; $i < $n ; $i ++) echo $arr [$i] . " " ; // This code is contributed by // Rajput-Ji ?> Output Array before deletion 10 50 30 40 20

Array after deletion 10 50 40 20 Time Complexity: O(N) Auxiliary Space: O(1) Comment Article Tags: Article Tags: Searching DSA Arrays