|  |  |
| --- | --- |
| **Check number exists in array in C++** | |
| #include <iostream>  using namespace std;  int array11(int nums[], int index, int length) {      if (index >= length) {          return 0;      }      int small = array11(nums, index + 1, length);      if (nums[index] == 11) {          return 1 + small;      } else {          return small;      }  }  int main() {      int arr[] = {1, 11, 3, 11, 11, 11};      int length = sizeof(arr) / sizeof(arr[0]);      cout << array11(arr, 0, length) << endl;      return 0;  } | 🧾 ****Input**** arr = {1, 11, 3, 11, 11, 11} 🔍 ****Function Call Tree**** array11(arr, 0, 6)  → nums[0] == 1 → skip  → array11(arr, 1, 6)  → nums[1] == 11 → count +1  → array11(arr, 2, 6)  → nums[2] == 3 → skip  → array11(arr, 3, 6)  → nums[3] == 11 → count +1  → array11(arr, 4, 6)  → nums[4] == 11 → count +1  → array11(arr, 5, 6)  → nums[5] == 11 → count +1  → array11(arr, 6, 6)  → index >= length → return 0 📋 ****Dry Run Table****  | **Call** | **index** | **nums[index]** | **Matches 11?** | **Return Value** | | --- | --- | --- | --- | --- | | array11(arr, 0, 6) | 0 | 1 | ❌ | 0 + 4 = 4 | | array11(arr, 1, 6) | 1 | 11 | ✅ | 1 + 3 = 4 | | array11(arr, 2, 6) | 2 | 3 | ❌ | 0 + 3 = 3 | | array11(arr, 3, 6) | 3 | 11 | ✅ | 1 + 2 = 3 | | array11(arr, 4, 6) | 4 | 11 | ✅ | 1 + 1 = 2 | | array11(arr, 5, 6) | 5 | 11 | ✅ | 1 + 0 = 1 | | array11(arr, 6, 6) | 6 | N/A | N/A | 0 |   **Output**  4 |
| Output:- 4 | |