





Solution Review: Delete an Element at a Specific Index

Let's go over the solution review of the challenge given in the previous lesson.

We'll cover the following

- Solution
- Explanation
 - delete_element function

Solution#

Press the **RUN** button and see the output!

```
12
13
   // delete_element function
    void delete_element(int *&arr, int size, int index) {
      // Declare new array dynamically
15
      int * new_arr = new int[size - 1];
16
17
      // Traverse array
      for (int i = 0; i < size - 1; i++) {
19
        //
20
        if (i == index || i > index) {
          new_arr[i] = arr[i + 1];
21
        }
22
23
        else {
24
          // Copy elements in new array
25
          new_arr[i] = arr[i];
26
27
        }
28
29
      // Free memory pointed out by arr
```

```
30
       delete[] arr;
       // Pointer arr will point to new_arr
31
                                                                 €€}}
32
       arr = new_arr;
33
       //return arr;
34
    }
35
36
    // main function
37
    int main() {
38
39
       // Initialize variables
 \triangleright
                                                                X
Output
                                                                           1.09s
 0 1 2 3 4
 0 1 2 4
```

Explanation#

To delete the element at the given index, we copy the elements before the given index in a new array. However, when we reach the given index, we left shift the rest of the values in a new array. In this way, we delete the element at the given index.

delete_element function

The delete_element function takes a pointer to the int array in its input parameters. It also takes the values for size and index.

First, declare a new array dynamically of a size equal to size-1 as we will be deleting one element. Traverse the original array. If the index i is less than the index to be deleted, simply copy the elements from the original array to a new array (Line No. 25). When i becomes equal to index, ignore



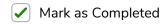
the index element and fill each element of a new array with the next element of the original array. Free the memory pointed out by arr and point arr to new_arr.



Challenge 2: Delete an Element at a S...



Challenge 3: Calculate Mean and Stan...



!) Report an Issue