Python Basic - 1: Exercise-73 with Solution

Write a Python program that removes duplicate elements from a given array of numbers so that each element appears only once and returns the new length of the array.

**Sample Solution**:

**Python Code:**

# Define a function to remove duplicates from a sorted list

def remove\_duplicates(nums):

# Iterate through the list in reverse order

for i in range(len(nums)-1, 0, -1):

# Check if the current element is equal to the previous one

if nums[i] == nums[i-1]:

# If equal, delete the previous element to remove the duplicate

del nums[i-1]

# Return the length of the modified list

return len(nums)

# Test the function with two different lists and print the results

print(remove\_duplicates([0,0,1,1,2,2,3,3,4,4,4])) # [0, 1, 2, 3, 4], Length: 5

print(remove\_duplicates([1, 2, 2, 3, 4, 4])) # [1, 2, 3, 4], Length: 4

Copy

Sample Output:

5

4

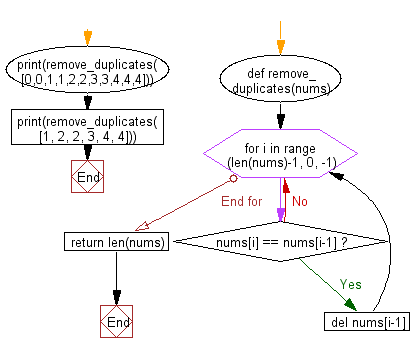
**Explanation:**

Here is a breakdown of the above Python code:

* First the function "remove\_duplicates" takes a sorted list (nums) as input.
* It iterates through the list in reverse order using a 'for' loop.
* Inside the loop, it compares each element with its previous one to identify duplicates.
* If a duplicate is found, it deletes the previous element using 'del' to remove the duplicate.
* The function returns the length of the modified list after removing duplicates.
* Test the function with two different lists and print the modified lists along with their lengths.

**Visual Presentation:**

**Flowchart:**



The job is simple. We need to take a list, with duplicate elements in it and generate another list that only contains the element without the duplicates in them.

**Examples:**

Input : [2, 4, 10, 20, 5, 2, 20, 4]

Output : [2, 4, 10, 20, 5]

Input : [28, 42, 28, 16, 90, 42, 42, 28]

Output : [28, 42, 16, 90]

We can use [not in on list](https://www.geeksforgeeks.org/list-methods-in-python-set-1-in-not-in-len-min-max/) to find out the duplicate items. We create a result list and insert only those that are not already not in.

* Python3

|  |
| --- |
| # Python code to remove duplicate elements  def Remove(duplicate):      final\_list = []      for num in duplicate:          if num not in final\_list:              final\_list.append(num)      return final\_list    # Driver Code  duplicate = [2, 4, 10, 20, 5, 2, 20, 4]  print(Remove(duplicate)) |

**Output:**

[2, 4, 10, 20, 5]

**Easy Implementation:**

A quick way to do the above using set data structure from the python standard library (Python 3.x implementation is given below)

* Python3

|  |
| --- |
| duplicate = [2, 4, 10, 20, 5, 2, 20, 4]  print(list(set(duplicate))) |

**Output:**

[2, 4, 10, 20, 5]

## Method 2: Using Dictionary/hashmap

### Approach:

1. Create a dictionary and by default keep the count of every element to zero, using the default dict.
2. If the count of elements is ZERO, increment the value in the dictionary and continue.
3. If the count of element is greater than zero, Then remove the element from the given list using the remove() method.

You can read more about [default dict here.](https://www.geeksforgeeks.org/defaultdict-in-python/)

* Python3

|  |
| --- |
| from collections import defaultdict      def default\_val():      return 0      # dict : maintain count of each element. with default value of key is 0  mydict = defaultdict(default\_val)    # LIST  l = [1, 2, 3, 2, 6, 3, 5, 3, 7, 8]    for i in l:      # if the element already present in the array, remove the element.      if mydict[i] == 1:          l.remove(i)      # If the elements appears first time keep it count as 1      else:          mydict[i] = 1    # printing the final array  print(l) |

**Output**

[1, 2, 6, 5, 3, 7, 8]

### Method: Using dict.fromkeys()

* Python3

|  |
| --- |
| input\_list = [1, 2, 3, 2, 6, 3, 5, 3, 7, 8]  mylist = list(dict.fromkeys(input\_list))  print(mylist) |

**Output**

[1, 2, 3, 6, 5, 7, 8]

***Time complexity:****Average-case time complexity: O(n); n is the size of the array*

*Worst-case time complexity: O(); n is the size of the array*

***Auxiliary Space:****O(n); n is the size of the array*

### ****Method: Using list comprehension:****

* Python

|  |
| --- |
| list1 = [1, 2, 3, 2, 6, 3, 5, 3, 7, 8]  mylist = [ list1[i] for i in range(len(list1)) if list1.index(list1[i]) == i]  print(mylist) |

**Output:**

[1, 2, 3, 6, 5, 7, 8]

**Method: Using Counter() function**

* Python3

|  |
| --- |
| # Python code to remove duplicate elements  from collections import Counter  # Driver Code  duplicate = [2, 4, 10, 20, 5, 2, 20, 4]  unique = Counter(duplicate)  print(list(unique.keys())) |

**Output**

[2, 4, 10, 20, 5]

Time Complexity: O(N)

Auxiliary Space : O(N)

**Method: Using operator.countOf() method**

* Python3

|  |
| --- |
| import operator as op  # Python code to remove duplicate elements      def Remove(duplicate):      final\_list = []      for num in duplicate:          if op.countOf(final\_list, num) == 0:              final\_list.append(num)      return final\_list      # Driver Code  duplicate = [2, 4, 10, 20, 5, 2, 20, 4]  print(Remove(duplicate)) |

**Output**

[2, 4, 10, 20, 5]

Time Complexity: O(N)

Auxiliary Space : O(N)

Don't miss your chance to ride the wave of the data revolution! Every industry is scaling new heights by tapping into the power of data. Sharpen your skills and become a part of the hottest trend in the 21st century.