

## 1.Remove element

main.py	Output
<pre>1 def remove(nums, val): 2     k=0 3     for num in nums: 4         if num!=val: 5             nums[k]=num 6             k+=1 7             print(num) 8     print("no.of elements:") 9     return k 10 print(remove([1,2,3,4,3],3))</pre>	<pre>1 2 4 no.of elemnts: 3  === Code Execution Successful ===</pre>

Time complexity: $O(n)$

## 2.Permutations sequence

main.py	Output
<pre>1 from itertools import permutations 2 l=list(permutations(range(1,5))) 3 print(l[9])</pre>	<pre>(2, 3, 4, 1)  === Code Execution Successful ===</pre>

Time complexity: $O(n \log n)$

## 3.Maximum sub array

main.py	Output
<pre>1 def max_subarray_sum(nums): 2     max_sum = current_sum = nums[0] 3 4     for i in range(1, len(nums)): 5         current_sum = max(nums[i], current_sum + nums[i]) 6         max_sum = max(max_sum, current_sum) 7     return max_sum 8 nums = [-2, 1, -3, 4, -1, 3, 1, -5, 7] 9 print(max_subarray_sum(nums))</pre>	<pre>9  === Code Execution Successful ===</pre>

Time complexity:  $O(n^2)$

## 4.combination sum

main.py	Output
<pre>1 def find_pairs(lst, K): 2     result = [] 3     for i in range(len(lst)): 4         for j in range(i + 1, len(lst)): 5             if lst[i] + lst[j] == K: 6                 result.append((lst[i], lst[j])) 7     return result 8 lst=[1, 5, 3, 7, 9] 9 K=12 10 print(find_pairs(lst,K))</pre>	<pre>[(5, 7), (3, 9)]  === Code Execution Successful ===</pre>

Time complexity:  $O(n \log n)$