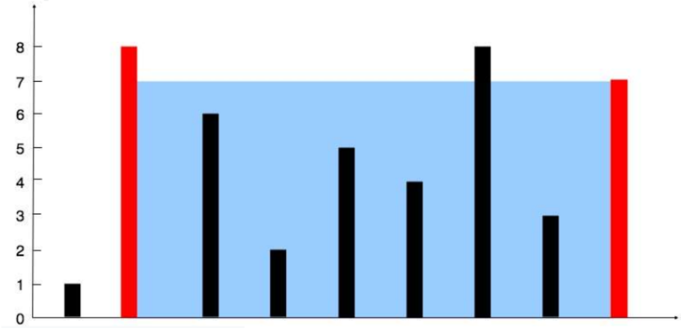
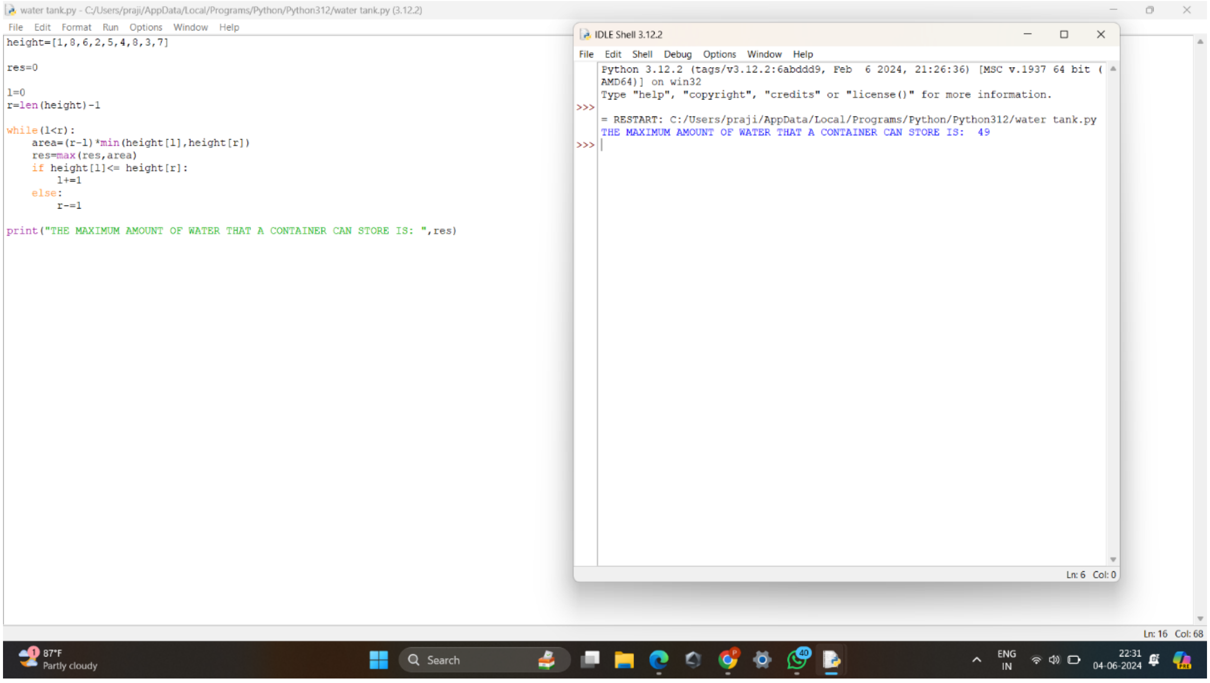
11 .Container With Most Water You are given an integer array height of length n. There are n vertical lines drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]). Find two lines that together with the x-axis form a container, such that the container contains the most water. Return the maximum amount of water a container can store. Notice that you may not slant the container.



Example 2:

Input: height = [1,8,6,2,5,4,8,3,7] Output: 49 Explanation: The above vertical lines are represented by array [1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the container can contain is 49.


water tank.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/water tank.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 16 Col: 68
87°F
Partly cloudy
Search
ENG
IN
22:31
04-06-2024

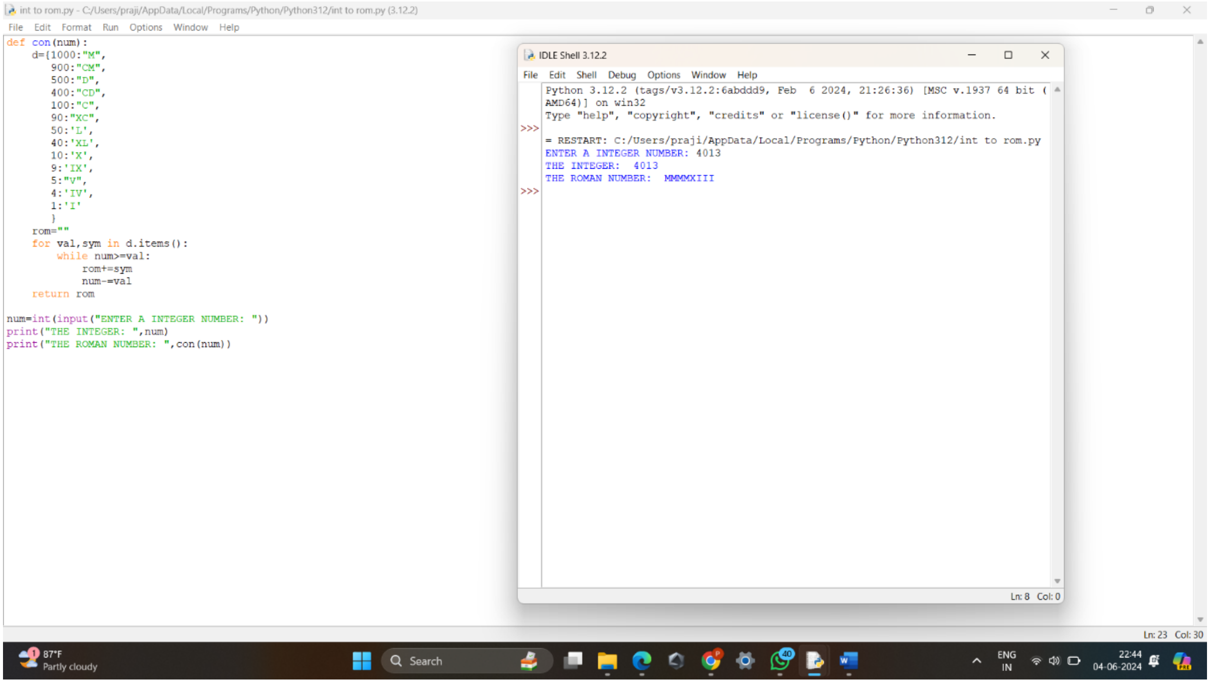
```
height=[1,8,6,2,5,4,8,3,7]

res=0

l=0
r=len(height)-1

while(l<r):
    area=(r-l)*min(height[l],height[r])
    res=max(res,area)
    if height[l]<= height[r]:
        l+=1
    else:
        r-=1

print("THE MAXIMUM AMOUNT OF WATER THAT A CONTAINER CAN STORE IS: ",res)
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/water tank.py
THE MAXIMUM AMOUNT OF WATER THAT A CONTAINER CAN STORE IS:  49
>>> 
```

Time: O(n)

12. Integer to Roman Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M. Symbol Value I 1 V 5 X 10 L 50 C 100 D 500 M 1000 For example, 2 is written as II in Roman numeral, just two one's added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II. Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used: ● I can be placed before V (5) and X (10) to make 4 and 9. ● X can be placed before L (50) and C (100) to make 40 and 90. ● C can be placed before D (500) and M (1000) to make 400 and 900. Given an integer, convert it to a roman numeral.

Example 1: Input: num = 3 Output: "III" Explanation: 3 is represented as 3 ones.


int to rom.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/int to rom.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 8 Col: 0
Ln: 23 Col: 30
87°F
Partly cloudy
Search
ENG
IN
22:44
04-06-2024

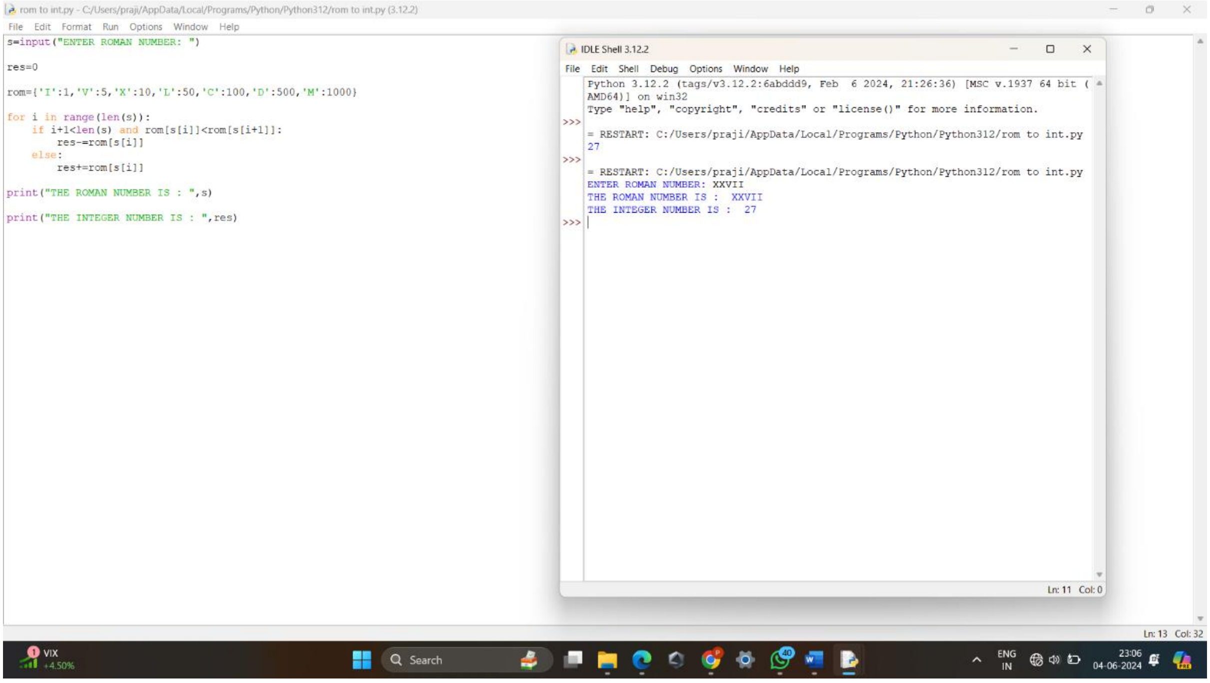
```
def con(num):
    d={1000:"M",
       900:"CM",
       500:"D",
       400:"CD",
       100:"C",
       90:"XC",
       50:'L',
       40:'XL',
       10:'X',
       9:'IX',
       5:"V",
       4:'IV',
       1:'I'
       }
    rom=""
    for val,sym in d.items():
        while num>=val:
            rom+=sym
            num-=val
    return rom

num=int(input("ENTER A INTEGER NUMBER: "))
print("THE INTEGER: ",num)
print("THE ROMAN NUMBER: ",con(num))
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/int to rom.py
ENTER A INTEGER NUMBER: 4013
THE INTEGER:  4013
THE ROMAN NUMBER:  MMMMXIII
>>>
```

Time: O(n)

13. Roman to Integer Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M. Symbol Value I 1 V 5 X 10 L 50 C 100 D 500 M 1000 For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II. Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used: ● I can be placed before V (5) and X (10) to make 4 and 9. ● X can be placed before L (50) and C (100) to make 40 and 90. ● C can be placed before D (500) and M (1000) to make 400 and 900. Given a roman numeral, convert it to an integer.

Example 1: Input: s = "III" Output: 3 Explanation: III = 3.


rom to int.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/rom to int.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 11 Col: 0
Ln: 13 Col: 32

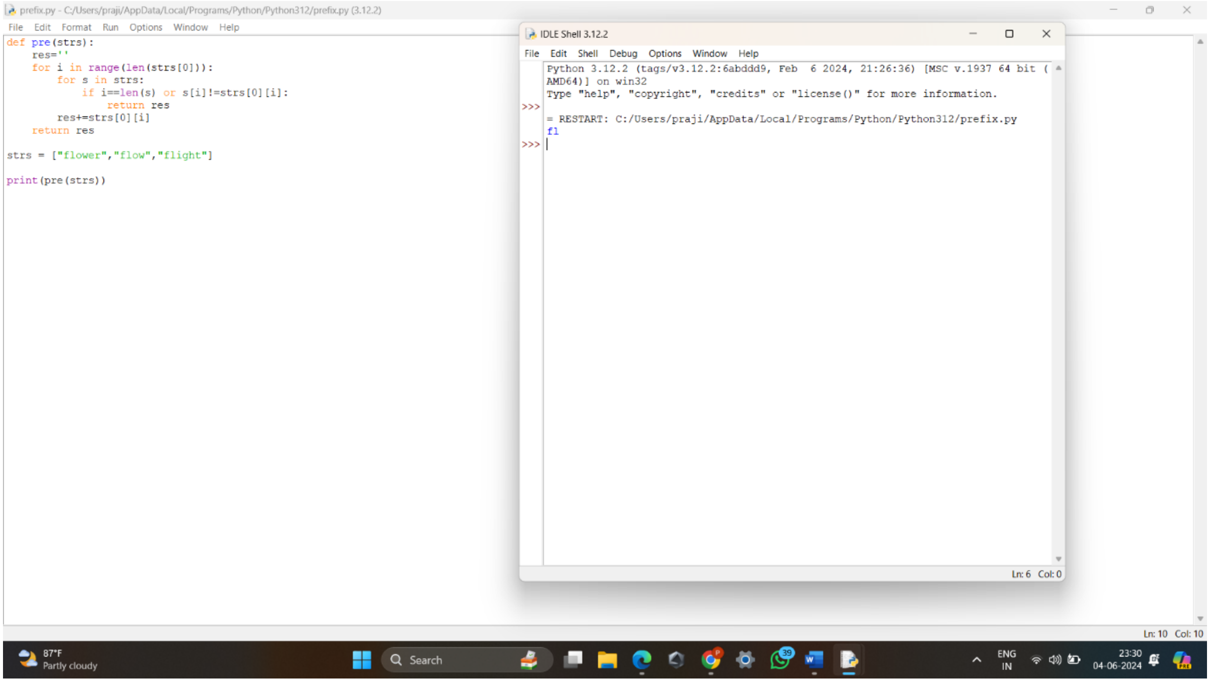
```
s=input("ENTER ROMAN NUMBER: ")

res=0

rom={'I':1,'V':5,'X':10,'L':50,'C':100,'D':500,'M':1000}

for i in range(len(s)):
    if i+1<len(s) and rom[s[i]]<rom[s[i+1]]:
        res-=rom[s[i]]
    else:
        res+=rom[s[i]]

print("THE ROMAN NUMBER IS : ",s)

print("THE INTEGER NUMBER IS : ",res)
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/rom to int.py
27
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/rom to int.py
ENTER ROMAN NUMBER: XXVII
THE ROMAN NUMBER IS :  XXVII
THE INTEGER NUMBER IS :  27
>>>
```

Time:O(n)

14. Longest Common Prefix Write a function to find the longest common prefix string amongst an array of strings. If there is no common prefix, return an empty string "".

Example 1: Input: strs = ["flower","flow","flight"] Output: "fl"


prefix.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/prefix.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 10 Col: 10
87°F
Partly cloudy
Search
ENG
IN
23:30
04-06-2024

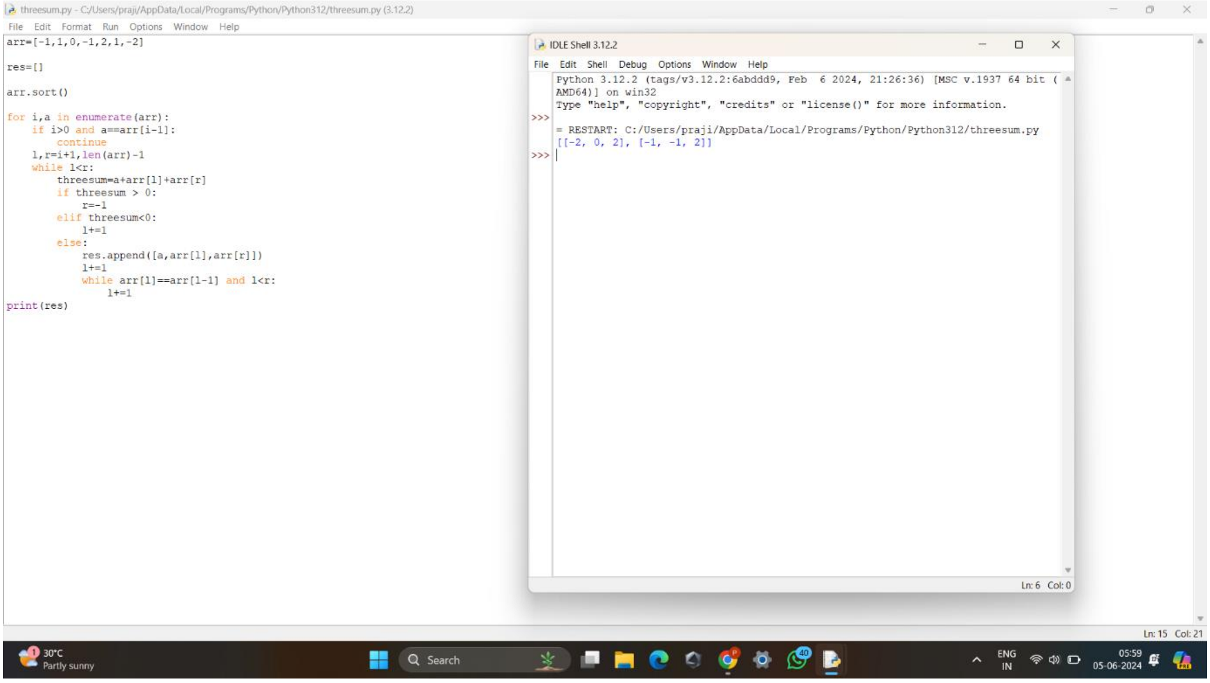
```
def pre(strs):
    res=''
    for i in range(len(strs[0])):
        for s in strs:
            if i==len(s) or s[i]!=strs[0][i]:
                return res
        res+=strs[0][i]
    return res

strs = ["flower","flow","flight"]

print(pre(strs))
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/prefix.py
fl
>>> 
```

Time:O(n^2)

15. 3 Sum Given an integer array nums, return all the triplets [nums[i], nums[j], nums[k]] such that i != j, i != k, and j != k, and nums[i] + nums[j] + nums[k] == 0. Notice that the solution set must not contain duplicate triplets.

Example 1: Input: nums = [-1,0,1,2,-1,-4] Output: [[-1,-1,2],[-1,0,1]] Explanation: nums[0] + nums[1] + nums[2] = (-1) + 0 + 1 = 0. nums[1] + nums[2] + nums[4] = 0 + 1 + (-1) = 0. nums[0] + nums[3] + nums[4] = (-1) + 2 + (-1) = 0. The distinct triplets are [-1,0,1] and [-1,-1,2]. Notice that the order of the output and the order of the triplets does not matter


threesum.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/threesum.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 15 Col: 21
30°C
Partly sunny
Search
ENG
IN
05:59
05-06-2024

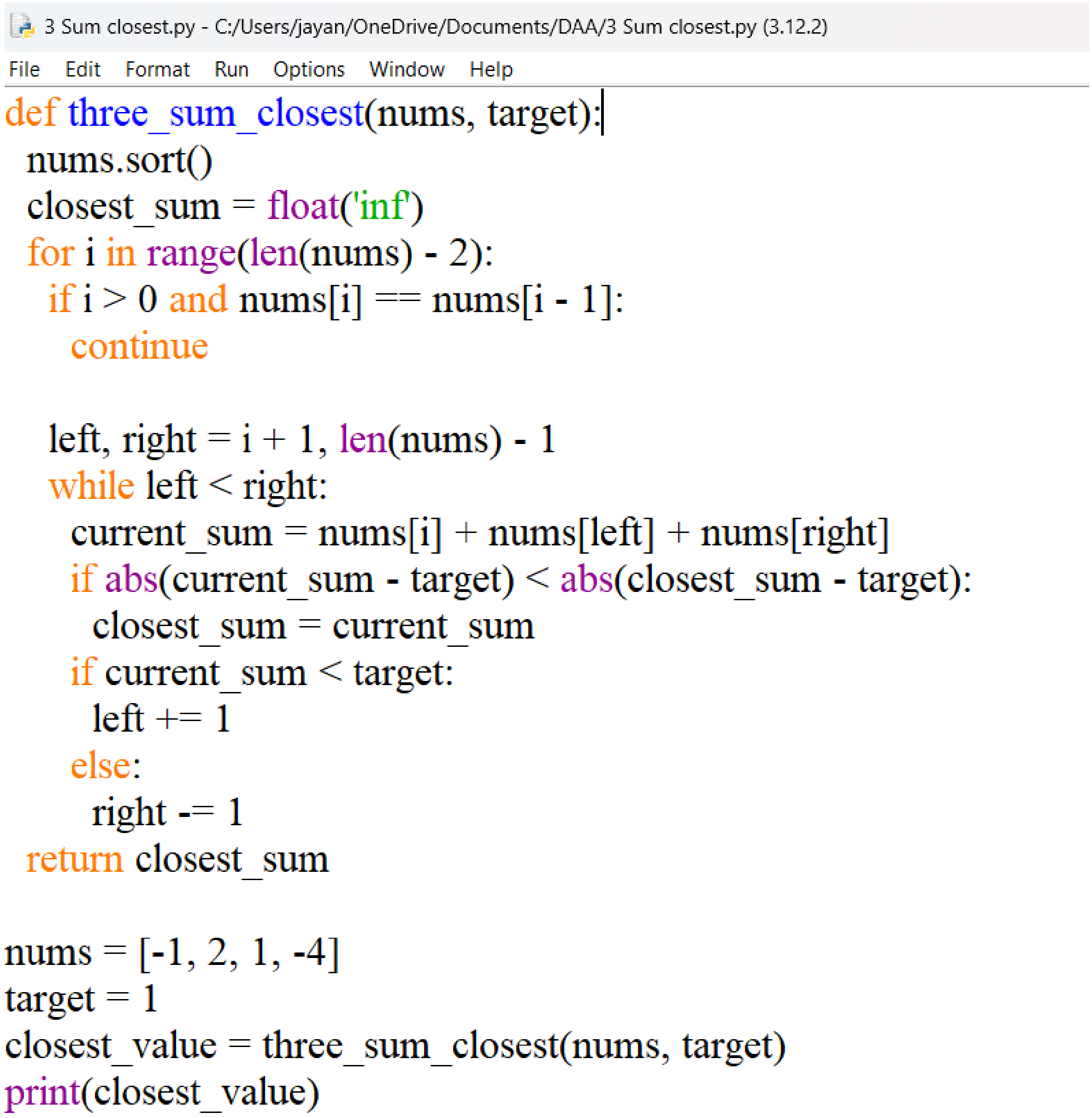
```
arr=[-1,1,0,-1,2,1,-2]

res=[]

arr.sort()

for i,a in enumerate(arr):
    if i>0 and a==arr[i-1]:
        continue
    l,r=i+1,len(arr)-1
    while l<r:
        threesum=a+arr[l]+arr[r]
        if threesum > 0:
            r=-1
        elif threesum<0:
            l+=1
        else:
            res.append([a,arr[l],arr[r]])
            l+=1
            while arr[l]==arr[l-1] and l<r:
                l+=1
print(res)
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/threesum.py
[[-2, 0, 2], [-1, -1, 2]]
>>>
```

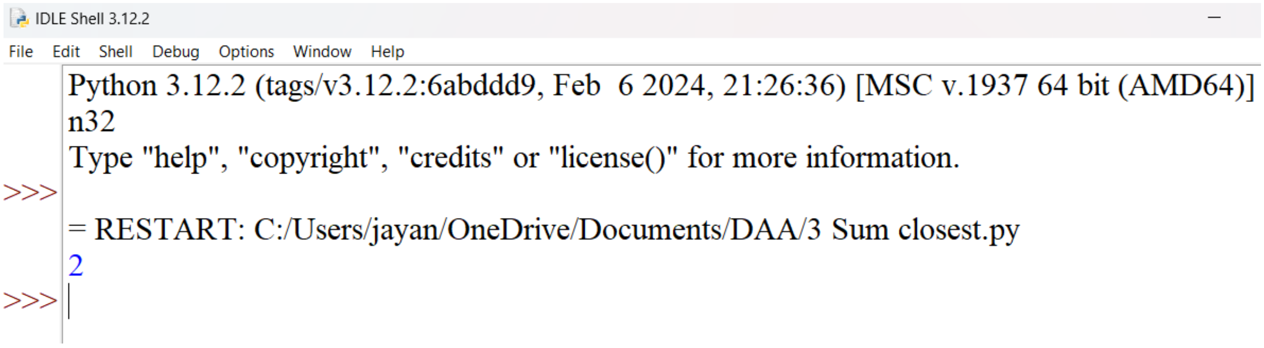
Time: O(n)

16. 3 Sum Closest Given an integer array nums of length n and an integer target, find three integers in nums such that the sum is closest to target. Return the sum of the three integers. You may assume that each input would have exactly one solution.

Example 1: Input: nums = [-1,2,1,-4], target = 1 Output: 2 Explanation: The sum that is closest to the target is 2. (-1 + 2 + 1 = 2).

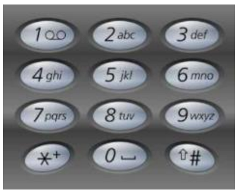

3 Sum closest.py - C:/Users/jayan/OneDrive/Documents/DAA/3 Sum closest.py (3.12.2)
File Edit Format Run Options Window Help

```python
def three_sum_closest(nums, target):
    nums.sort()
    closest_sum = float('inf')
    for i in range(len(nums) - 2):
        if i > 0 and nums[i] == nums[i - 1]:
            continue

        left, right = i + 1, len(nums) - 1
        while left < right:
            current_sum = nums[i] + nums[left] + nums[right]
            if abs(current_sum - target) < abs(closest_sum - target):
                closest_sum = current_sum
            if current_sum < target:
                left += 1
            else:
                right -= 1
    return closest_sum

nums = [-1, 2, 1, -4]
target = 1
closest_value = three_sum_closest(nums, target)
print(closest_value)
```

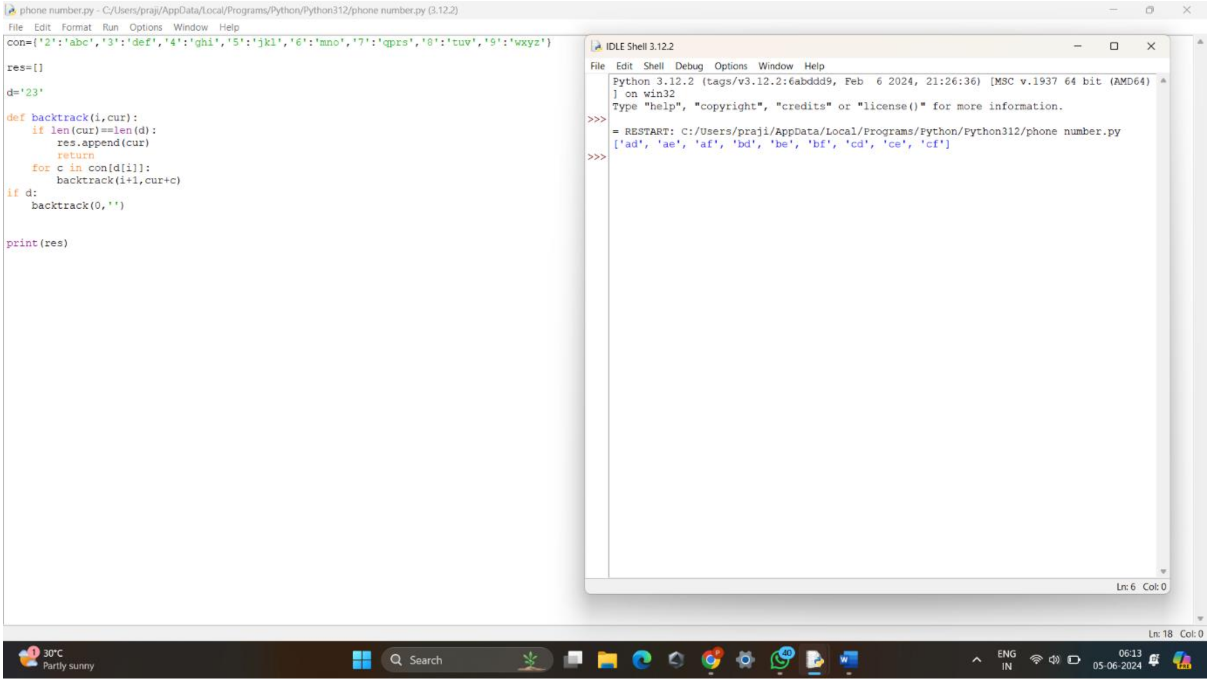


```
    Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)]
    n32
    Type "help", "copyright", "credits" or "license()" for more information.
>>> 
    = RESTART: C:/Users/jayan/OneDrive/Documents/DAA/3 Sum closest.py
    2
>>> 
```

17. Letter Combinations of a Phone Number Given a string containing digits from 2-9 inclusive, return all possible letter combinations that the number could represent. Return the answer in any order. A mapping of digits to letters (just like on the telephone buttons) is given below. Note that 1 does not map to any letters.



Example 1: Input: digits = "23" Output: ["ad","ae","af","bd","be","bf","cd","ce","cf"]


phone number.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/phone number.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 18 Col: 0
30°C
Partly sunny
Search
ENG
IN
06:13
05-06-2024

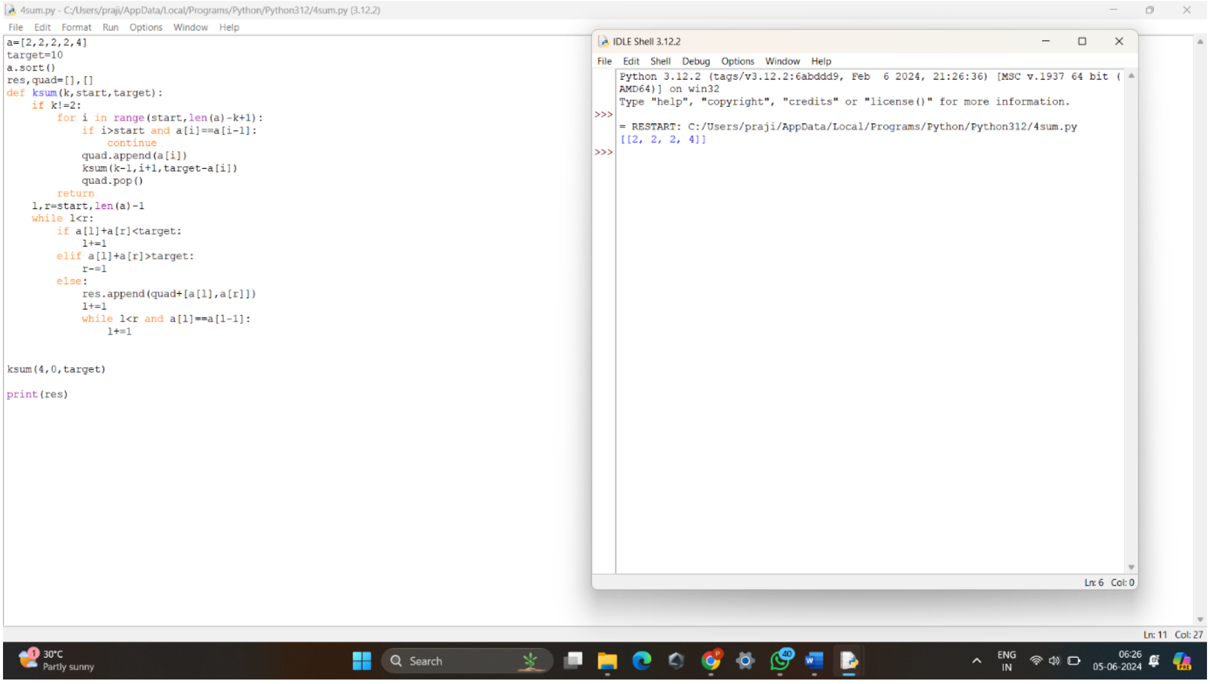
```python
con={'2':'abc','3':'def','4':'ghi','5':'jkl','6':'mno','7':'qprs','8':'tuv','9':'wxyz'}

res=[]

d='23'

def backtrack(i,cur):
    if len(cur)==len(d):
        res.append(cur)
        return
    for c in con[d[i]]:
        backtrack(i+1,cur+c)
if d:
    backtrack(0,'')


print(res)
```

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)
] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/phone number.py
['ad', 'ae', 'af', 'bd', 'be', 'bf', 'cd', 'ce', 'cf']
>>>
```

Time:O(4^n)

18. 4 Sum Given an array nums of n integers, return an array of all the unique quadruplets [nums[a], nums[b], nums[c], nums[d]] such that: ● 0 <= a, b, c, d < n ● a, b, c, and d are distinct. ● nums[a] + nums[b] + nums[c] + nums[d] == target You may return the answer in any order.

Example 1: Input: nums = [1,0,-1,0,-2,2], target = 0 Output: [[-2,-1,1,2],[-2,0,0,2],[-1,0,0,1]]


4sum.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/4sum.py (3.12.2)
File Edit Format Run Options Window Help
IDLE Shell 3.12.2
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 11 Col: 27
30°C
Partly sunny
Search
ENG
IN
06:26
05-06-2024

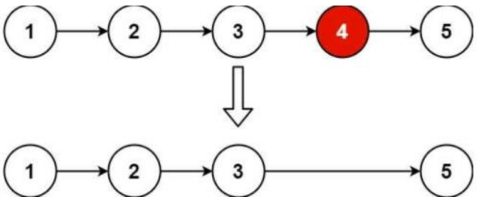
```
a=[2,2,2,2,4]
target=10
a.sort()
res,quad=[],[]
def ksum(k,start,target):
    if k!=2:
        for i in range(start,len(a)-k+1):
            if i>start and a[i]==a[i-1]:
                continue
            quad.append(a[i])
            ksum(k-1,i+1,target-a[i])
            quad.pop()
        return
    l,r=start,len(a)-1
    while l<r:
        if a[l]+a[r]<target:
            l+=1
        elif a[l]+a[r]>target:
            r-=1
        else:
            res.append(quad+[a[l],a[r]])
            l+=1
            while l<r and a[l]==a[l-1]:
                l+=1


ksum(4,0,target)

print(res)
```

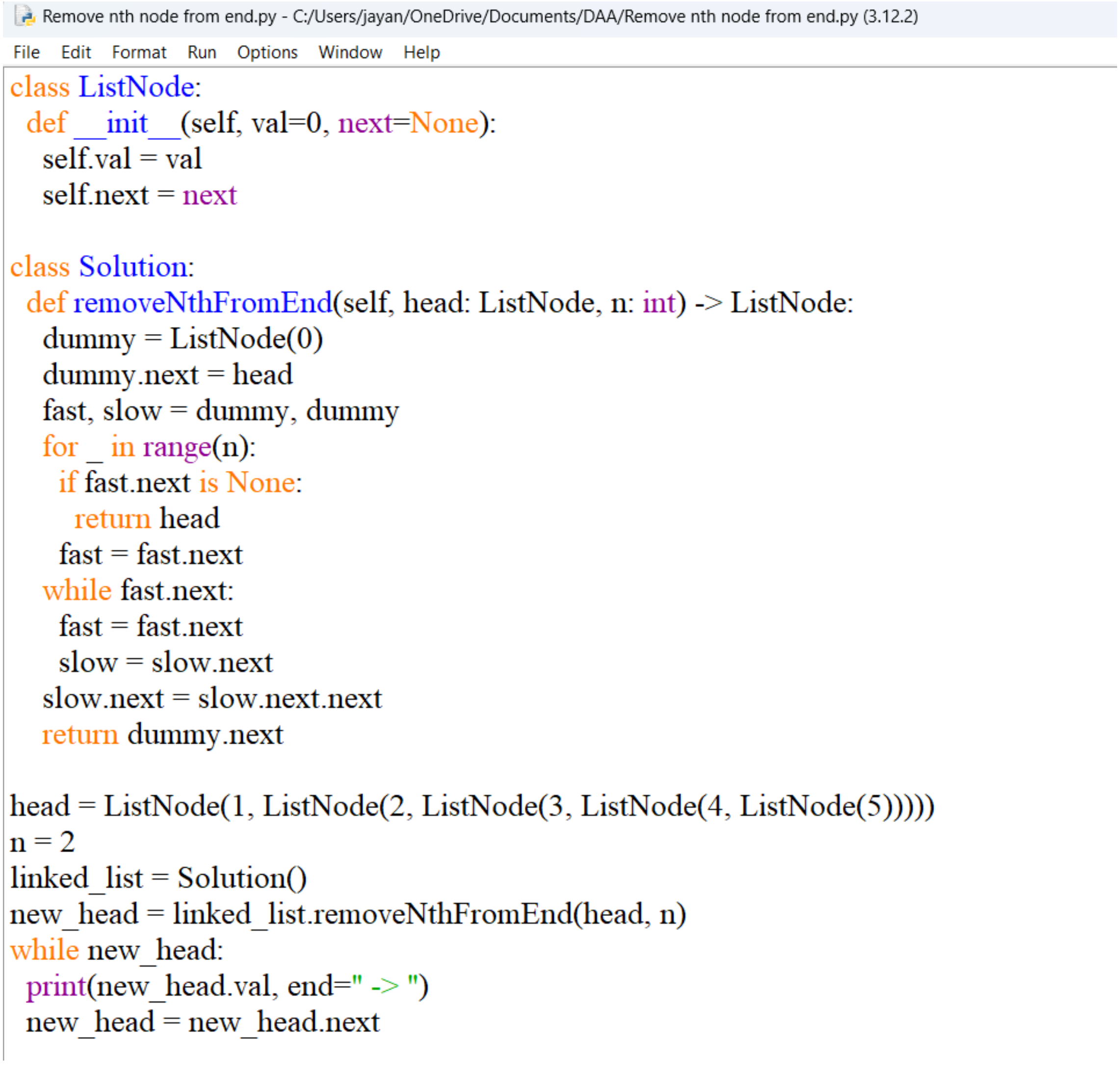
```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/4sum.py
[[2, 2, 2, 4]]
>>>
```

Time: O(n)

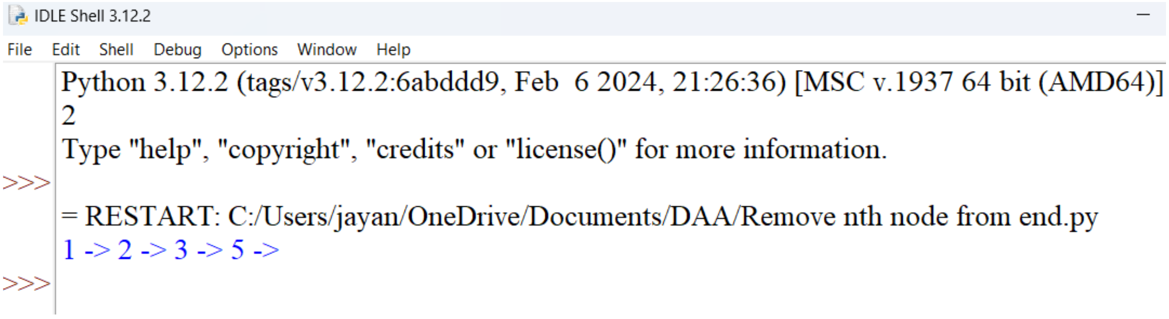
19. Remove Nth Node From End of List Given the head of a linked list, remove the nth node from the end of the list and return its head.

Example 1: Input: head = [1,2,3,4,5], n = 2 Output: [1,2,3,5]




Remove nth node from end.py - C:/Users/jayan/OneDrive/Documents/DAA/Remove nth node from end.py (3.12.2)
File Edit Format Run Options Window Help

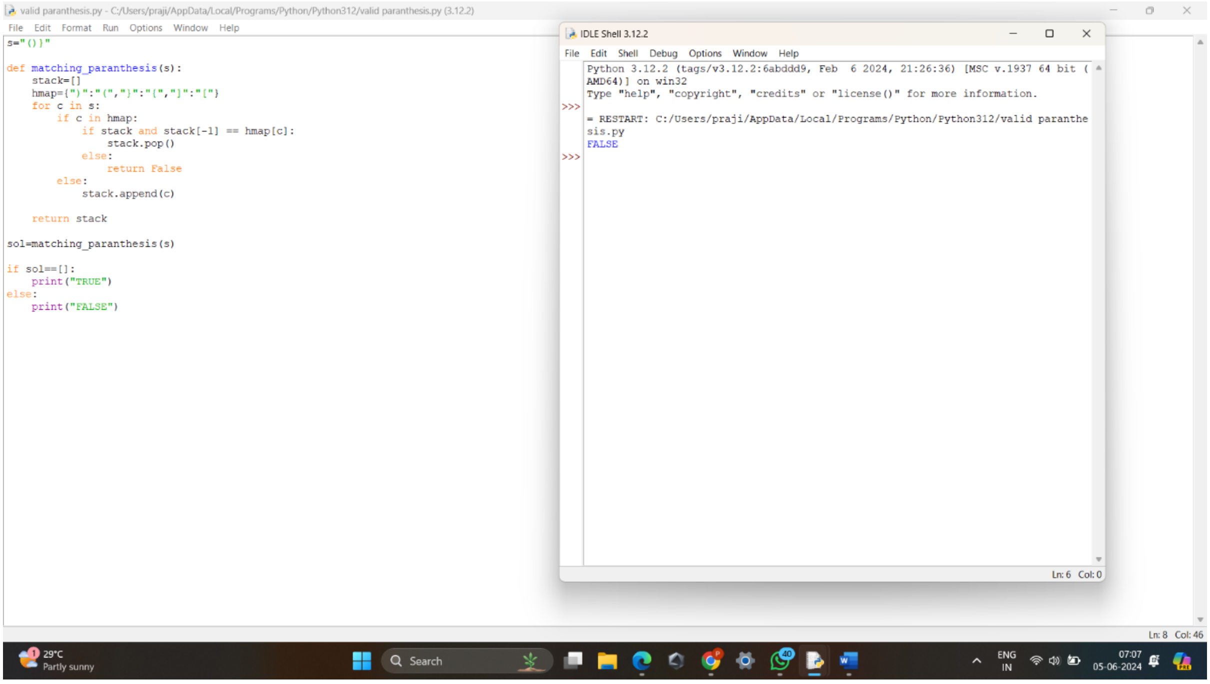
```python
class ListNode:
  def __init__(self, val=0, next=None):
    self.val = val
    self.next = next

class Solution:
  def removeNthFromEnd(self, head: ListNode, n: int) -> ListNode:
    dummy = ListNode(0)
    dummy.next = head
    fast, slow = dummy, dummy
    for _ in range(n):
      if fast.next is None:
        return head
      fast = fast.next
    while fast.next:
      fast = fast.next
      slow = slow.next
    slow.next = slow.next.next
    return dummy.next

head = ListNode(1, ListNode(2, ListNode(3, ListNode(4, ListNode(5)))))
n = 2
linked_list = Solution()
new_head = linked_list.removeNthFromEnd(head, n)
while new_head:
  print(new_head.val, end=" -> ")
  new_head = new_head.next
```



```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (AMD64)]
2
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/jayan/OneDrive/Documents/DAA/Remove nth node from end.py
1 -> 2 -> 3 -> 5 ->
>>>
```

20. Valid Parentheses Given a string s containing just the characters '(', ')', '{', '}', '[' and ']', determine if the input string is valid. An input string is valid if: 1. Open brackets must be closed by the same type of brackets. 2. Open brackets must be closed in the correct order. 3. Every close bracket has a corresponding open bracket of the same type.

Example 1: Input: s = "()" Output: true



valid paranthesis.py - C:/Users/praji/AppData/Local/Programs/Python/Python312/valid paranthesis.py (3.12.2)

```
s="()}"

def matching_paranthesis(s):
    stack=[]
    hmap={")":"(","}":"{","]":"["}
    for c in s:
        if c in hmap:
            if stack and stack[-1] == hmap[c]:
                stack.pop()
            else:
                return False
        else:
            stack.append(c)

    return stack

sol=matching_paranthesis(s)

if sol==[]:
    print("TRUE")
else:
    print("FALSE")
```

IDLE Shell 3.12.2

```
Python 3.12.2 (tags/v3.12.2:6abddd9, Feb  6 2024, 21:26:36) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
= RESTART: C:/Users/praji/AppData/Local/Programs/Python/Python312/valid paranthe
sis.py
FALSE
>>> 
```

File Edit Format Run Options Window Help
File Edit Shell Debug Options Window Help
Ln: 6 Col: 0
Ln: 8 Col: 46
29°C
Partly sunny
Search
ENG
IN
07:07
05-06-2024