

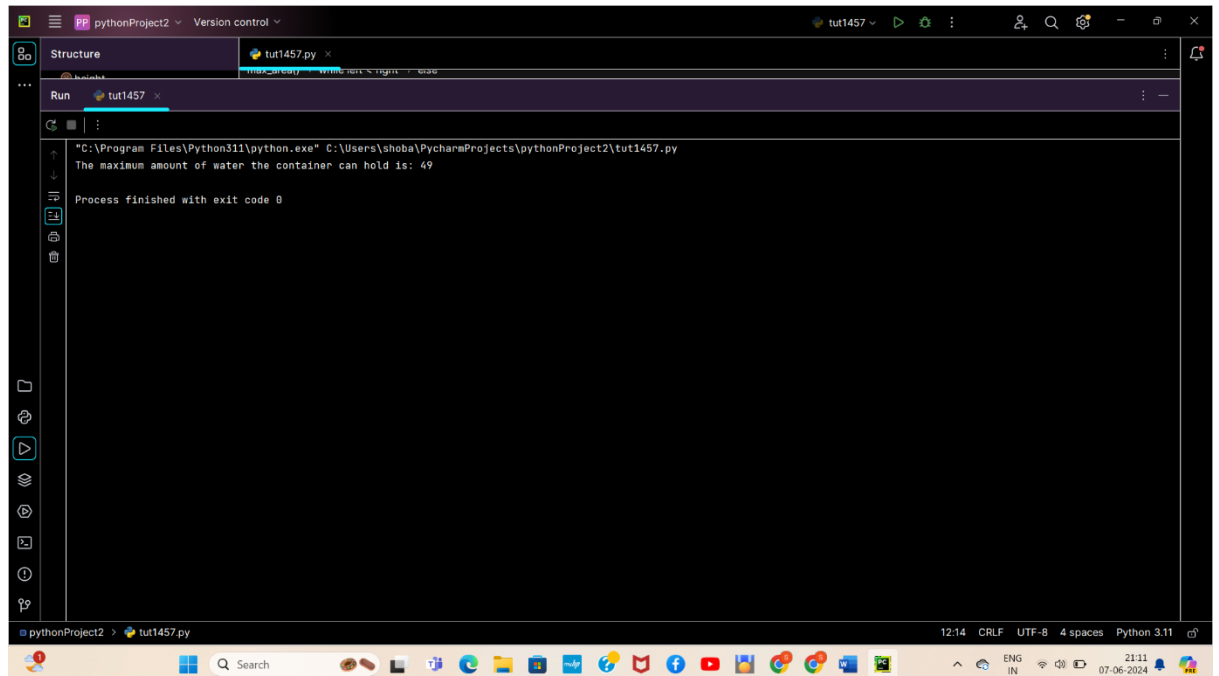
Assignment 2

1. Container With Most Water.

Program:

```
def max_area(height):
    left = 0
    right = len(height) - 1
    max_water = 0
    while left < right:
        width = right - left
        min_height = min(height[left], height[right])
        current_water = width * min_height
        max_water = max(max_water, current_water)
        if height[left] < height[right]:
            left += 1
        else:
            right -= 1
    return max_water
height = [1,8,6,2,5,4,8,3,7]
print("The maximum amount of water the container can hold is:",
max_area(height))
```

Output:

A screenshot of a Python IDE window titled 'pythonProject2'. The editor shows the code for the 'max_area' function and its execution with the input list [1, 8, 6, 2, 5, 4, 8, 3, 7]. The 'Run' panel at the bottom shows the output: 'The maximum amount of water the container can hold is: 49'. The status bar at the bottom indicates the file is 'tut1457.py', the encoding is 'UTF-8', and the Python version is '3.11'.

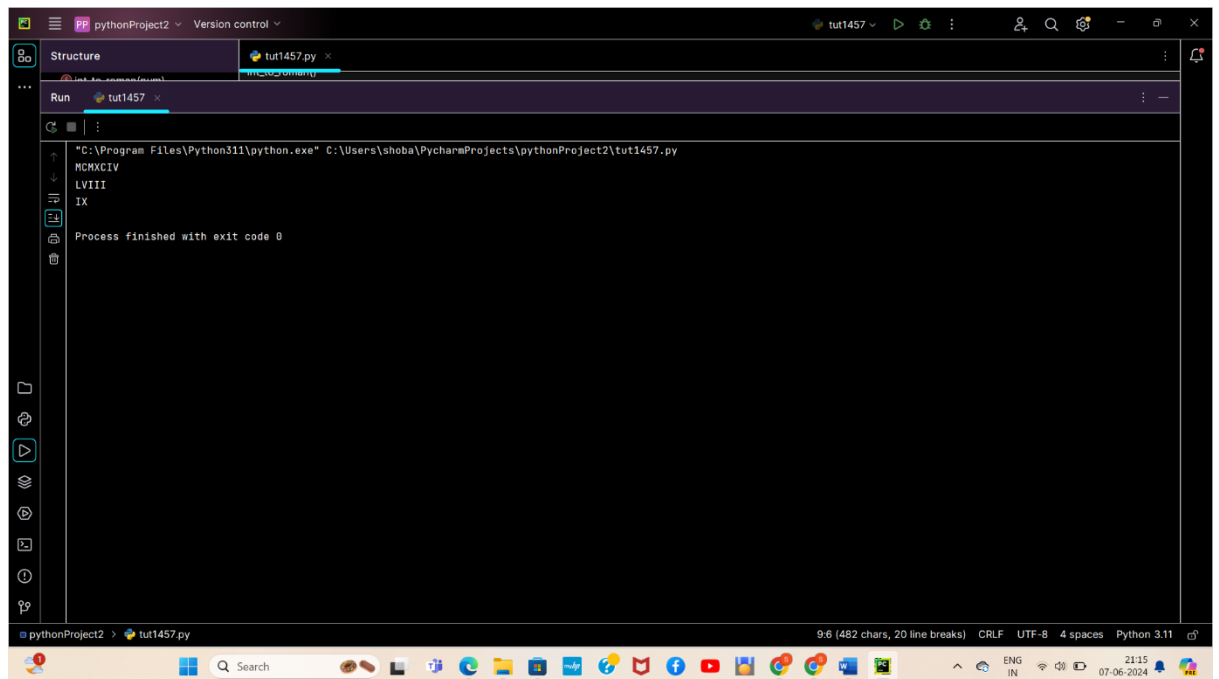
```
pythonProject2 > tut1457.py
The maximum amount of water the container can hold is: 49
Process finished with exit code 0
```

2. Integer to Roman.

Program:

```
def int_to_roman(num):
    val = [
        1000, 900, 500, 400, 100, 90, 50, 40,
        10, 9, 5, 4, 1
    ]
    syms = [
        "M", "CM", "D", "CD", "C", "XC", "L",
        "XL", "X", "IX", "V", "IV", "I"
    ]
    roman_numeral = ""
    i = 0
    while num > 0:
        for _ in range(num // val[i]):
            roman_numeral += syms[i]
            num -= val[i]
        i += 1
    return roman_numeral
print(int_to_roman(1994))
print(int_to_roman(58))
print(int_to_roman(9))
```

Output:



The screenshot shows a Python IDE with a file named 'tut1457.py'. The code defines a function 'int_to_roman' that converts integers to Roman numerals. The function uses two lists: 'val' for the values and 'syms' for the corresponding Roman numeral symbols. It processes the input number by repeatedly subtracting the largest possible value and appending the corresponding symbol. The output window shows the results of the function calls: 'MCMXCIV' for 1994, 'LVIII' for 58, and 'IX' for 9. The process finished with exit code 0.

3. Roman to Integer.

Program:

```
def roman_to_int(s):
    roman_values = {
        'I': 1,
```

```

        'V': 5,
        'X': 10,
        'L': 50,
        'C': 100,
        'D': 500,
        'M': 1000
    }

    total = 0
    prev_value = 0

    for char in reversed(s):
        current_value = roman_values[char]

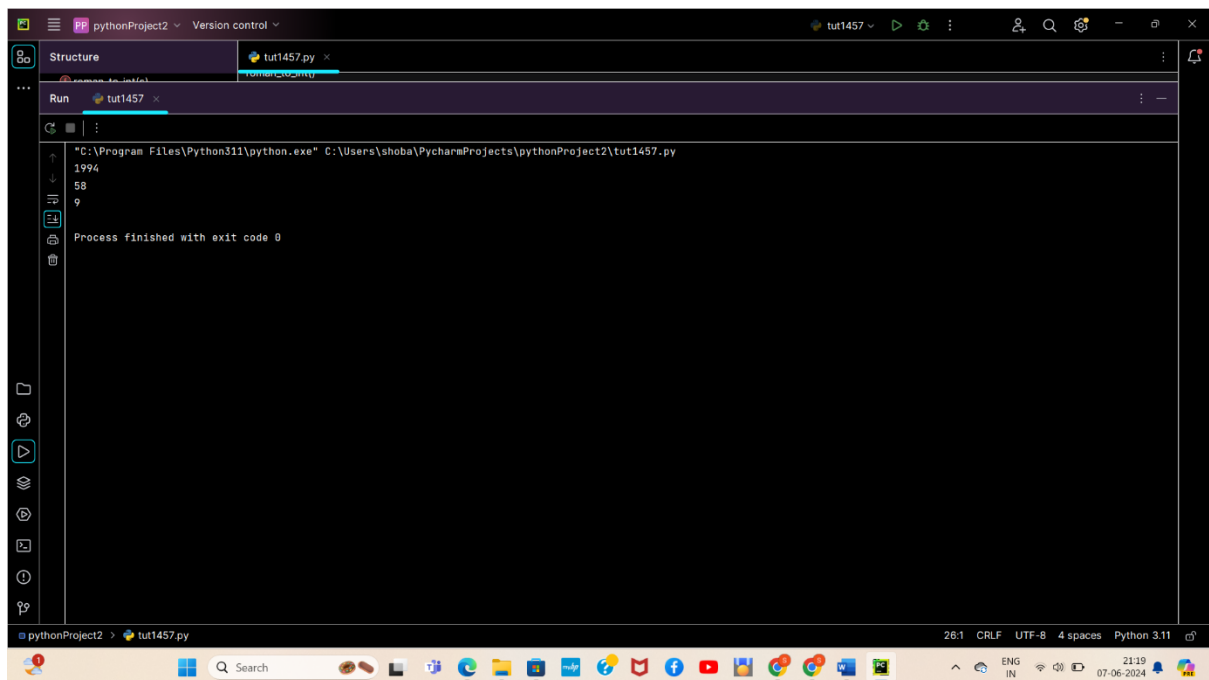
        if current_value < prev_value:
            total -= current_value
        else:
            total += current_value

        prev_value = current_value

    return total
print(roman_to_int("MCMXCIV"))
print(roman_to_int("LVIII"))
print(roman_to_int("IX"))

```

Output:



The screenshot shows the PyCharm IDE interface. The top toolbar includes icons for running and debugging. The 'Run' tab is active, showing the command: `"C:\Program Files\Python311\python.exe" C:\Users\shoba\PycharmProjects\pythonProject2\tut1457.py`. The output in the console is: `1994`, `58`, `9`, and `Process finished with exit code 0`. The status bar at the bottom indicates the file encoding is UTF-8, 4 spaces, and Python 3.11.

4. Longest Common Prefix Program:

```

def longest_common_prefix(strs):
    if not strs:

```

```

        return ""

    strs.sort()

    first = strs[0]
    last = strs[-1]

    i = 0
    while i < len(first) and i < len(last) and first[i] == last[i]:
        i += 1

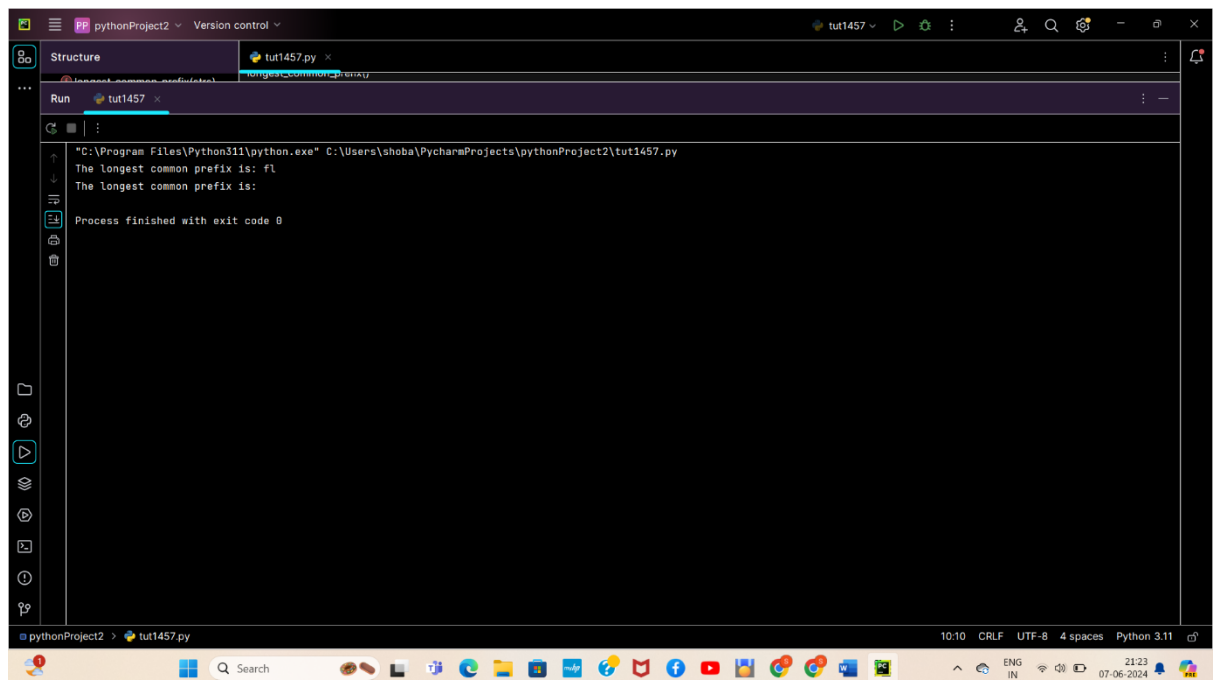
    return first[:i]

strs = ["flower", "flow", "flight"]
print("The longest common prefix is:", longest_common_prefix(strs))

strs = ["dog", "racecar", "car"]
print("The longest common prefix is:", longest_common_prefix(strs))

```

Output :



```

"C:\Program Files\Python311\python.exe" C:\Users\shoba\PycharmProjects\pythonProject2\tut1457.py
The longest common prefix is: fl
The longest common prefix is:
Process finished with exit code 0

```

5. 3Sum

Program:

```
def three_sum(nums):
    nums.sort()
    res = []

    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 current_sum == 0:
                res.append([nums[i], nums[left], nums[right]])

                while left < right and nums[left] == nums[left + 1]:
                    left += 1
                while left < right and nums[right] == nums[right - 1]:
                    right -= 1

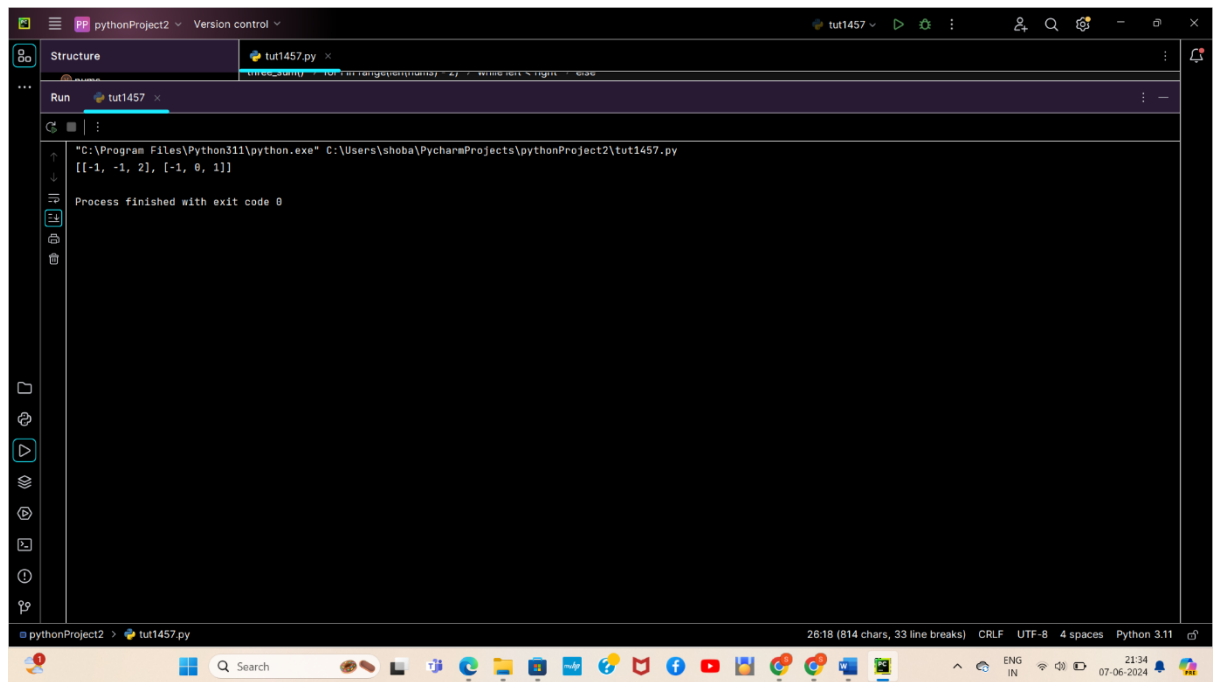
                left += 1
                right -= 1

            elif current_sum < 0:
                left += 1
            else:
                right -= 1

    return res

nums = [-1, 0, 1, 2, -1, -4]
print(three_sum(nums))
```

Output:



6. 3 Sum Closet Program:

```
def three_sum_closest(nums, target):
    nums.sort()
    closest_sum = float('inf')

    for i in range(len(nums) - 2):
        left, right = i + 1, len(nums) - 1
        while left < right:
            current_sum = nums[i] + nums[left] + nums[right]

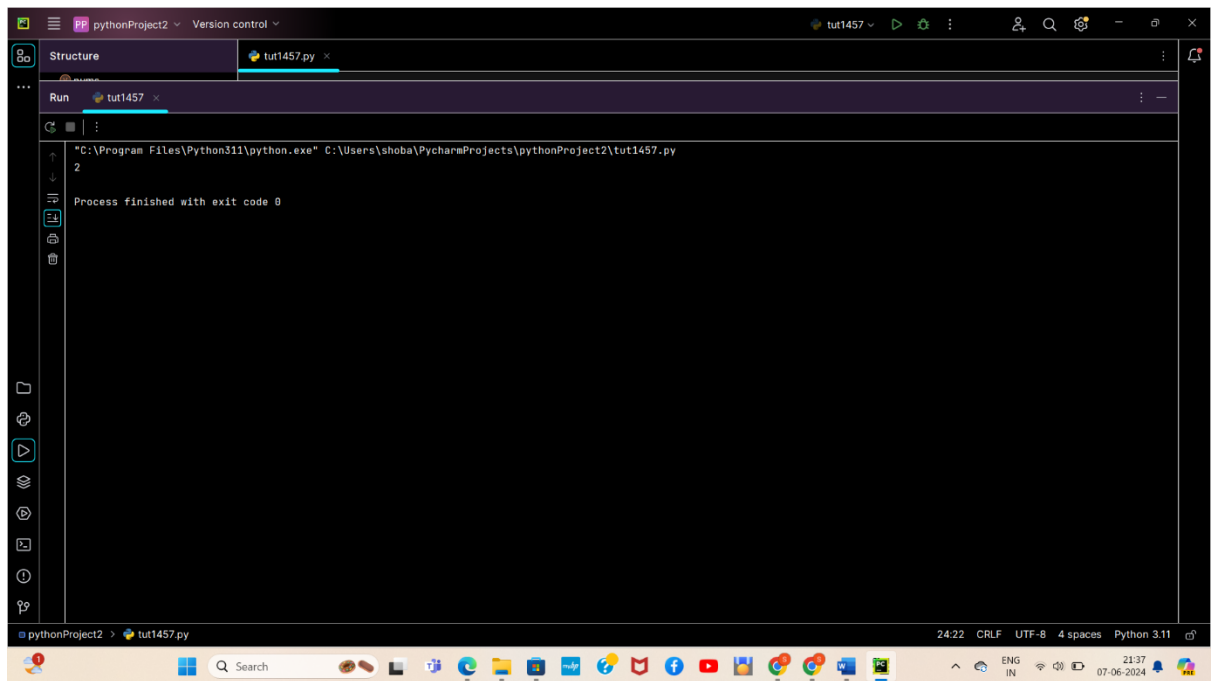
            if abs(current_sum - target) < abs(closest_sum - target):
                closest_sum = current_sum

            if current_sum == target:
                return current_sum
            elif current_sum < target:
                left += 1
            else:
                right -= 1

    return closest_sum

nums = [-1, 2, 1, -4]
target = 1
print(three_sum_closest(nums, target)) # Output: 2
```

Output:



7. Letter Combination of a phone Number.

Program:

```
def letter_combinations(digits):
    if not digits:
        return []

    phone_map = {
        '2': 'abc', '3': 'def', '4': 'ghi', '5': 'jkl',
        '6': 'mno', '7': 'pqrs', '8': 'tuv', '9': 'wxyz'
    }

    res = []

    def backtrack(index, path):
        if index == len(digits):
            res.append(''.join(path))
            return

        possible_letters = phone_map[digits[index]]

        for letter in possible_letters:
            path.append(letter)
            backtrack(index + 1, path)
            path.pop()

    backtrack(0, [])
```

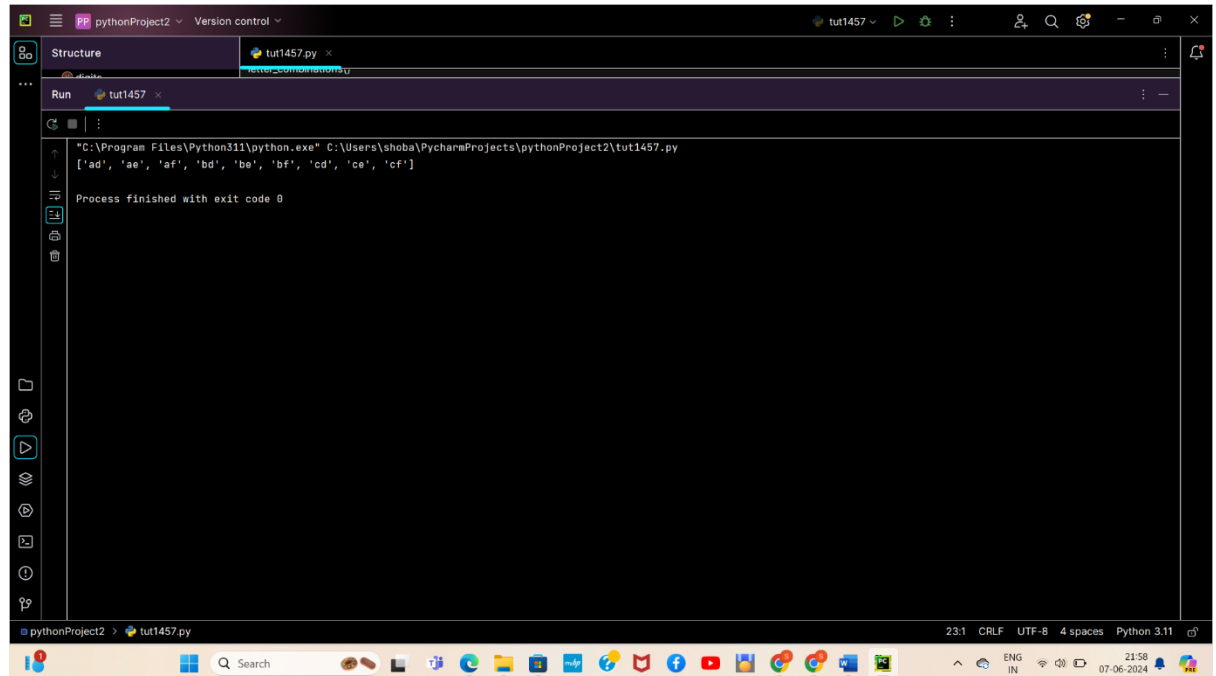
```

return res

digits = "23"
print(letter_combinations(digits))

```

Output:



8. Four Sum

Program:

```

def four_sum(nums, target):
    nums.sort()
    res = []
    length = len(nums)

    for i in range(length - 3):
        if i > 0 and nums[i] == nums[i - 1]:
            continue
        for j in range(i + 1, length - 2):
            if j > i + 1 and nums[j] == nums[j - 1]:
                continue
            left, right = j + 1, length - 1
            while left < right:
                current_sum = nums[i] + nums[j] + nums[left] + nums[right]
                if current_sum == target:
                    res.append([nums[i], nums[j], nums[left], nums[right]])
                    while left < right and nums[left] == nums[left + 1]:
                        left += 1
                    while left < right and nums[right] == nums[right - 1]:
                        right -= 1
                    left += 1

```



```

        right -= 1
    elif current_sum < target:
        left += 1
    else:
        right -= 1

    return res

nums = [1, 0, -1, 0, -2, 2]
target = 0
print(four_sum(nums, target))

```

Output:

The screenshot shows a PyCharm IDE window for a project named 'pythonProject2'. The 'Run' tab is active, displaying the execution of a Python script 'tut1457.py'. The console output shows the command: `"C:\Program Files\Python311\python.exe" C:\Users\shoba\PycharmProjects\pythonProject2\tut1457.py` followed by the output: `[[[-2, -1, 1, 2], [-2, 0, 0, 2], [-1, 0, 0, 1]]]`. Below the output, it states 'Process finished with exit code 0'. The status bar at the bottom indicates the file is 'tut1457.py', the encoding is 'UTF-8', and the Python version is '3.11'.

9. Remove Nth Node From End of List.

Program:

```
"C:\Program Files\Python311\python.exe" C:\Users\shoba\PycharmProjects\pythonProject2\tut1457.py
Original list:
[1, 2, 3, 4, 5]
List after removing nth node from end:
[1, 2, 3, 5]
Process finished with exit code 0
```

Output:

```
"C:\Program Files\Python311\python.exe" C:\Users\shoba\PycharmProjects\pythonProject2\tut1457.py
Original list:
[1, 2, 3, 4, 5]
List after removing nth node from end:
[1, 2, 3, 5]
Process finished with exit code 0
```

10. Valid Parentheses.

Program:

```

def is_valid(s):
    matching_bracket = {'(': ')', '[': ']', '{': '}'}
    stack = []

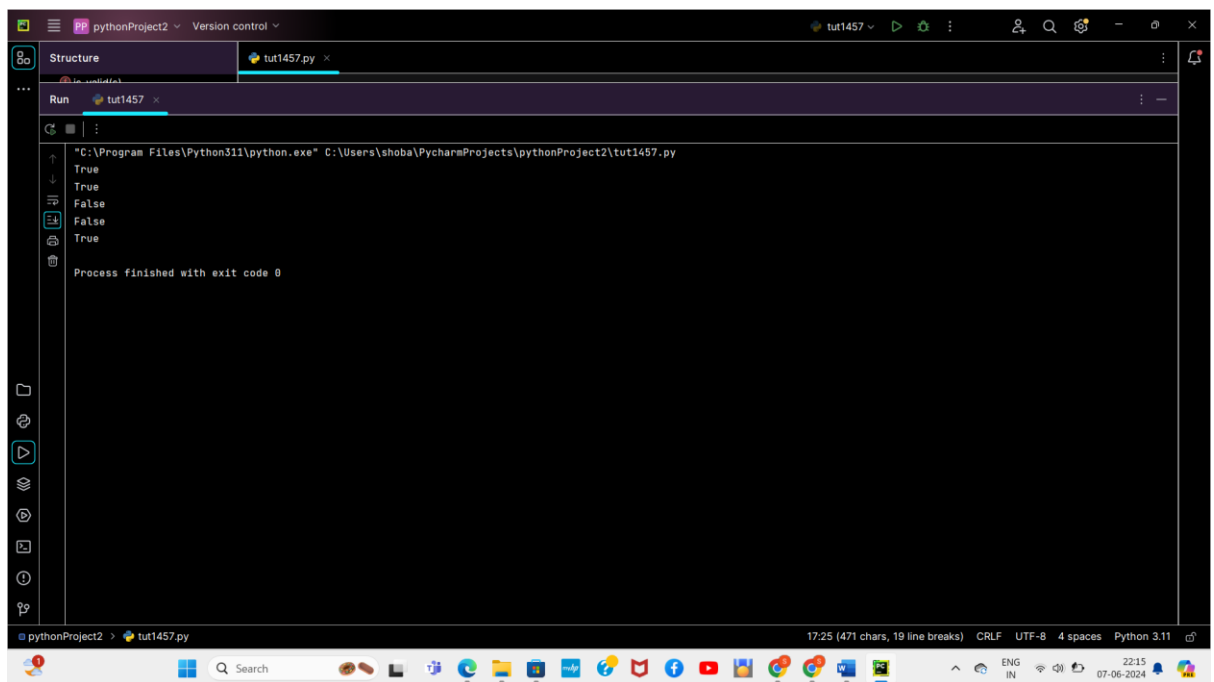
    for char in s:
        if char in matching_bracket:
            top_element = stack.pop() if stack else '#'
            if matching_bracket[char] != top_element:
                return False
        else:
            stack.append(char)

    return not stack

print(is_valid("()"))
print(is_valid("() [] {}"))
print(is_valid("[]"))
print(is_valid("( [])"))
print(is_valid("{} []"))

```

Output:



The screenshot shows the PyCharm IDE interface. The top toolbar includes icons for file operations, search, and running code. The 'Run' tab is active, showing the execution output of the script 'tut1457.py'. The output window displays the following results:

```

True
True
False
False
True

```

Below the output, it states "Process finished with exit code 0". The bottom status bar indicates the file is 'tut1457.py', the character count is 17:25 (471 chars, 19 line breaks), the encoding is CRLF UTF-8, and the Python version is 3.11.