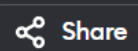


main.py



Share

Run

Output

```
1 def is_valid(s):
2     stack = []
3     mapping = {"(": ")", "{": "}", "[": "]"}
4
5     for char in s:
6         if char in mapping:
7             top_element = stack.pop() if stack else '#'
8             if mapping[char] != top_element:
9                 return False
10        else:
11            stack.append(char)
12
13    return not stack
14 print(is_valid("()"))
15 print(is_valid("{}[]{}"))
16 print(is_valid("[]"))
17
```

True

True

False

```
=== Code Execution Successful ===
```

```
1 def max_area(height):
2     max_water = 0
3     for i in range(len(height)):
4         for j in range(i+1, len(height)):
5             max_water = max(max_water, min(height[i], height[j]) * (j - i))
6     return max_water
7 height1 = [1, 8, 6, 2, 5, 4, 8, 3, 7]
8 print(max_area(height1)) # Output: 49
9
10 height2 = [1, 1]
11 print(max_area(height2)) # Output: 1
12
```

49

1

=== Code Execution Successful ===

```
1 def int_to_roman(num):
2     val = [1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1]
3     syms = ["M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV",
4             "I"]
5     roman_num = ''
6     i = 0
7     while num > 0:
8         for _ in range(num // val[i]):
9             roman_num += syms[i]
10            num -= val[i]
11        i += 1
12    return roman_num
13 num = 354
14 print(f"Roman numeral for {num} is: {int_to_roman(num)}")
```

Roman numeral for 354 is: CCCLIV

=== Code Execution Successful ===

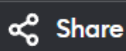
```
1 def roman_to_int(s: str) -> int:
2     roman_dict = {'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M':
      1000}
3     result = 0
4     prev_value = 0
5
6     for char in s:
7         value = roman_dict[char]
8         result += value
9         if prev_value < value:
10            result -= 2 * prev_value
11            prev_value = value
12
13     return result
14 input_roman_numeral = "III"
15 output_integer = roman_to_int(input_roman_numeral)
16 print(f"Input: {input_roman_numeral}")
17 print(f"Output: {output_integer}")
18
```

Input: III

Output: 3

=== Code Execution Successful ===

main.py



Share

Run

Output

```
1 def longest_common_prefix(strs):
2     if not strs:
3         return ""
4
5     prefix = ""
6     for i in range(len(min(strs))):
7         if all(s[i] == strs[0][i] for s in strs):
8             prefix += strs[0][i]
9         else:
10            break
11
12    return prefix
13 strs = ["flower", "flow", "flight"]
14 print(longest_common_prefix(strs)) # Output: "fl"
15
```

fl

=== Code Execution Successful ===

```
1 def three_sum(nums):
2     nums.sort()
3     result = []
4     for i in range(len(nums) - 2):
5         if i > 0 and nums[i] == nums[i - 1]:
6             continue
7         left, right = i + 1, len(nums) - 1
8         while left < right:
9             total = nums[i] + nums[left] + nums[right]
10            if total < 0:
11                left += 1
12            elif total > 0:
13                right -= 1
14            else:
15                result.append([nums[i], nums[left], nums[right]])
16                while left < right and nums[left] == nums[left + 1]:
17                    left += 1
18                while left < right and nums[right] == nums[right - 1]:
19                    right -= 1
20                left += 1
21                right -= 1
22     return result
23 nums = [-1, 0, 1, 2, -1, -4]
24 print(three_sum(nums))
25
```

[[-1, -1, 2], [-1, 0, 1]]

=== Code Execution Successful ===

```

1 def fourSum(nums, target):
2     nums.sort()
3     n = len(nums)
4     result = []
5
6     for i in range(n - 3):
7         if i > 0 and nums[i] == nums[i - 1]:
8             continue
9
10        for j in range(i + 1, n - 2):
11            if j > i + 1 and nums[j] == nums[j - 1]:
12                continue
13
14            left, right = j + 1, n - 1
15            while left < right:
16                total = nums[i] + nums[j] + nums[left] + nums[right]
17                if total == target:
18                    result.append([nums[i], nums[j], nums[left], nums[right]])
19                    while left < right and nums[left] == nums[left + 1]:
20                        left += 1
21                    while left < right and nums[right] == nums[right - 1]:
22                        right -= 1
23                    left += 1
24                    right -= 1
25                elif total < target:
26                    left += 1
27                else:
28                    right -= 1

```

```

[[-2, -1, 1, 2], [-2, 0, 0, 2], [-1, 0, 0, 1]]

```

```

=== Code Execution Successful ===

```