Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
from typing import List
def letterCombinations(digits: str) -> List[str]:
    if not digits:
        return []
    phone = {
        '2': 'abc',
        '3': 'def',
        '4': 'ghi',
        '5': 'jkl',
        '6': 'mno',
        '7': 'pqrs',
        '8': 'tuv',
        '9': 'wxyz'
    }
    def backtrack(index, path):
        if index == len(digits):
            combinations.append(''.join(path))
            return

        for letter in phone[digits[index]]:
            path.append(letter)
            backtrack(index + 1, path)
            path.pop()
    combinations = []
    backtrack(0, [])
    return combinations
digits = "23"
output = letterCombinations(digits)
```

input

```
['ad', 'ae', 'af', 'bd', 'be', 'bf', 'cd', 'ce', 'cf']


...Program finished with exit code 0
Press ENTER to exit console.
```

Run | Debug | Stop | Share | Save | {} Beautify | Language Python 3

main.py

```python
class Solution:
    def threeSumClosest(self, 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(target - current_sum) < abs(target - closest_sum):
                    closest_sum = current_sum
                if current_sum < target:
                    left += 1
                else:
                    right -= 1
            if closest_sum == target:
                break
        return closest_sum

# Example
nums = [-1, 2, 1, -4]
target = 1
solution = Solution()
output = solution.threeSumClosest(nums, target)
print(output)

```

input

```
2


...Program finished with exit code 0
Press ENTER to exit console.
```

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

def addTwoNumbers(l1, l2):
    dummy_head = ListNode(0)
    current = dummy_head
    carry = 0

    while l1 or l2 or carry:
        val1 = l1.val if l1 else 0
        val2 = l2.val if l2 else 0

        total = val1 + val2 + carry
        carry = total // 10
        current.next = ListNode(total % 10)
        current = current.next

        if l1:
            l1 = l1.next
        if l2:
            l2 = l2.next

    return dummy_head.next


def create_linked_list(lst):
```

input

```
7 -> 0 -> 8 -> None


...Program finished with exit code 0
Press ENTER to exit console.
```

```python
            l1 = l1.next
        if l2:
            l2 = l2.next

    return dummy_head.next


def create_linked_list(lst):
    dummy = ListNode(0)
    current = dummy
    for number in lst:
        current.next = ListNode(number)
        current = current.next
    return dummy.next


def print_linked_list(node):
    while node:
        print(node.val, end=" -> ")
        node = node.next
    print("None")

l1 = create_linked_list([2, 4, 3])
l2 = create_linked_list([5, 6, 4])
result = addTwoNumbers(l1, l2)
print_linked_list(result)
```

input

```
7 -> 0 -> 8 -> None


...Program finished with exit code 0
Press ENTER to exit console.
```

Run | Debug | Stop | Share | Save | { } Beautify

main.py

```python
def longestCommonPrefix(strs):
    if not strs:
        return ""

    shortest = min(strs, key=len)
    for i, char in enumerate(shortest):
        for other in strs:
            if other[i] != char:
                return shortest[:i]
    return shortest

print("Longest Common Prefix Example:", longestCommonPrefix(["flower", "flow", "flight"]))
```

input

```
Longest Common Prefix Example: fl


...Program finished with exit code 0
Press ENTER to exit console.
```

Run | Debug | Stop | Share | Save | {} Beautify

main.py

```python
def threeSum(nums):
    nums.sort()
    res = []
    for i, a in enumerate(nums):
        if i > 0 and a == nums[i - 1]:
            continue
        l, r = i + 1, len(nums) - 1
        while l < r:
            three_sum = a + nums[l] + nums[r]
            if three_sum > 0:
                r -= 1
            elif three_sum < 0:
                l += 1
            else:
                res.append([a, nums[l], nums[r]])
                l += 1
                while nums[l] == nums[l - 1] and l < r:
                    l += 1
    return res
print("3Sum Example:", threeSum([-1, 0, 1, 2, -1, -4]))
```

input

```
3Sum Example: [[-1, -1, 2], [-1, 0, 1]]


...Program finished with exit code 0
Press ENTER to exit console.
```

Run | Debug | Stop | Share | Save | { } Beautify

main.py

```python
def romanToInt(s):
    roman = {'I': 1, 'V': 5, 'X': 10, 'L': 50, 'C': 100, 'D': 500, 'M': 1000}
    total = 0

    for i in range(len(s)):
        if i > 0 and roman[s[i]] > roman[s[i - 1]]:
            total += roman[s[i]] - 2 * roman[s[i - 1]]
        else:
            total += roman[s[i]]

    return total

print("Roman to Integer Example:", romanToInt("MCMXCIV"))
```

input

```
Roman to Integer Example: 1994


...Program finished with exit code 0
Press ENTER to exit console.
```

Run | Debug | Stop | Share | Save | { } Beautify

main.py

```python
def maxArea(height):
    left, right = 0, len(height) - 1
    max_area = 0

    while left < right:
        max_area = max(max_area, min(height[left], height[right]) * (right - left))
        if height[left] < height[right]:
            left += 1
        else:
            right -= 1

    return max_area
height = [1, 8, 6, 2, 5, 4, 8, 3, 7]
print("Container With Most Water Example:", maxArea(height))

```

input

```
Container With Most Water Example: 49


...Program finished with exit code 0
Press ENTER to exit console.
```

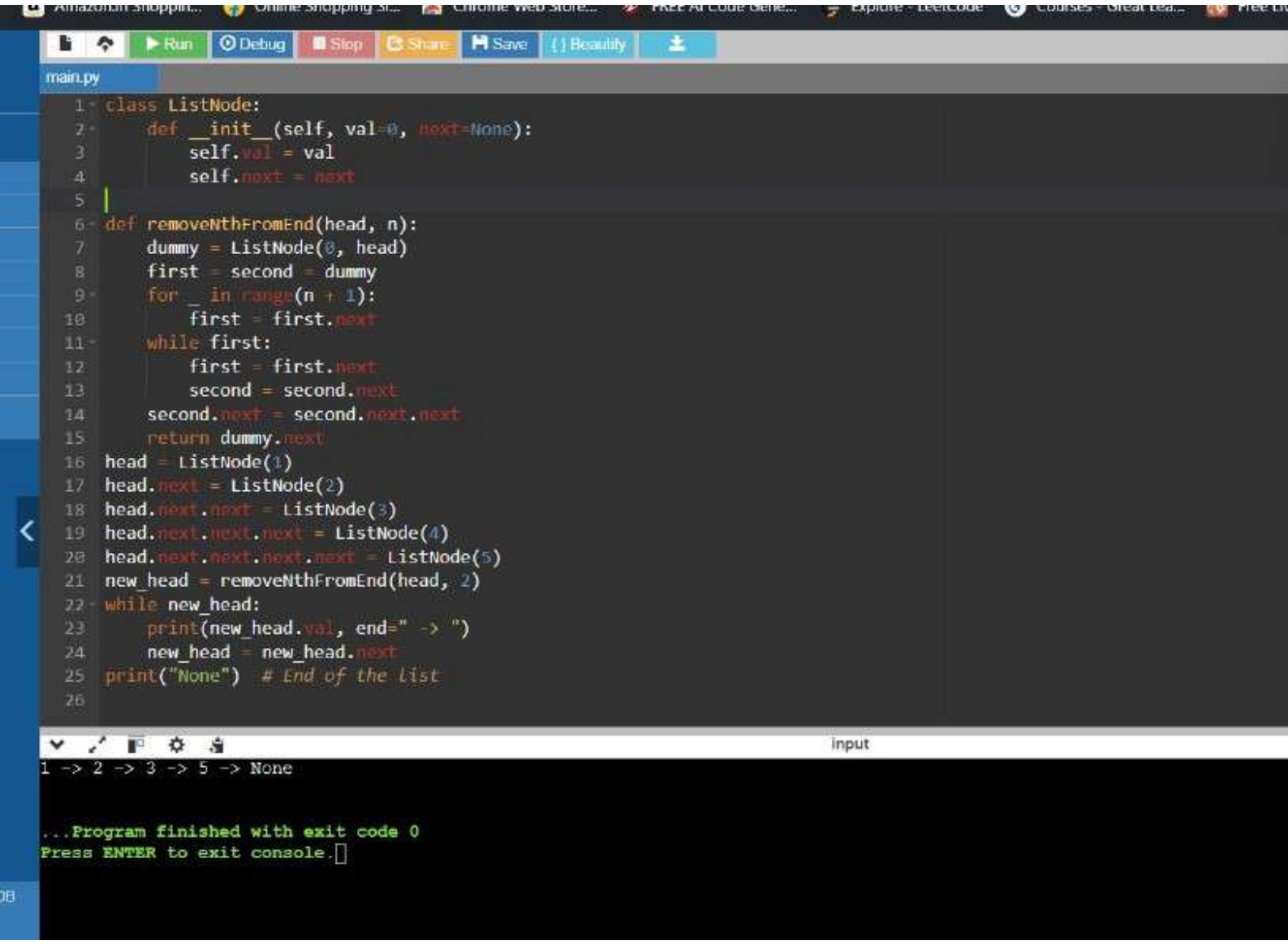
Run | Debug | Stop | Share | Save | { } Beautify

main.py

```python
def intToRoman(num):
    val = [
        1000, 900, 500, 400,
        100, 90, 50, 40,
        10, 9, 5, 4,
        1
        ]
    syb = [
        "M", "CM", "D", "CD",
        "C", "XC", "L", "XL",
        "X", "IX", "V", "IV",
        "I"
        ]
    roman_num = ''
    for i in range(len(val)):
        count = int(num / val[i])
        roman_num += syb[i] * count
        num -= val[i] * count
    return roman_num

print("Integer to Roman Example:", intToRoman(3549))
```

input

```
Integer to Roman Example: MMMDXLIX


...Program finished with exit code 0
Press ENTER to exit console.
```


Run
Debug
Stop
Share
Save
{} Beautify
main.py
class ListNode:
def __init__(self, val=0, next=None):
self.val = val
self.next = next
def removeNthFromEnd(head, n):
dummy = ListNode(0, head)
first = second = dummy
for _ in range(n + 1):
first = first.next
while first:
first = first.next
second = second.next
second.next = second.next.next
return dummy.next
head = ListNode(1)
head.next = ListNode(2)
head.next.next = ListNode(3)
head.next.next.next = ListNode(4)
head.next.next.next.next = ListNode(5)
new_head = removeNthFromEnd(head, 2)
while new_head:
print(new_head.val, end=" -> ")
new_head = new_head.next
print("None") # End of the list
input
1 -> 2 -> 3 -> 5 -> None
...Program finished with exit code 0
Press ENTER to exit console.

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

def addTwoNumbers(l1, l2):
    dummy_head = ListNode(0)
    current = dummy_head
    carry = 0

    while l1 or l2 or carry:
        val1 = l1.val if l1 else 0
        val2 = l2.val if l2 else 0

        total = val1 + val2 + carry
        carry = total // 10
        current.next = ListNode(total % 10)
        current = current.next

        if l1:
            l1 = l1.next
        if l2:
            l2 = l2.next

    return dummy_head.next


def create_linked_list(lst):
```

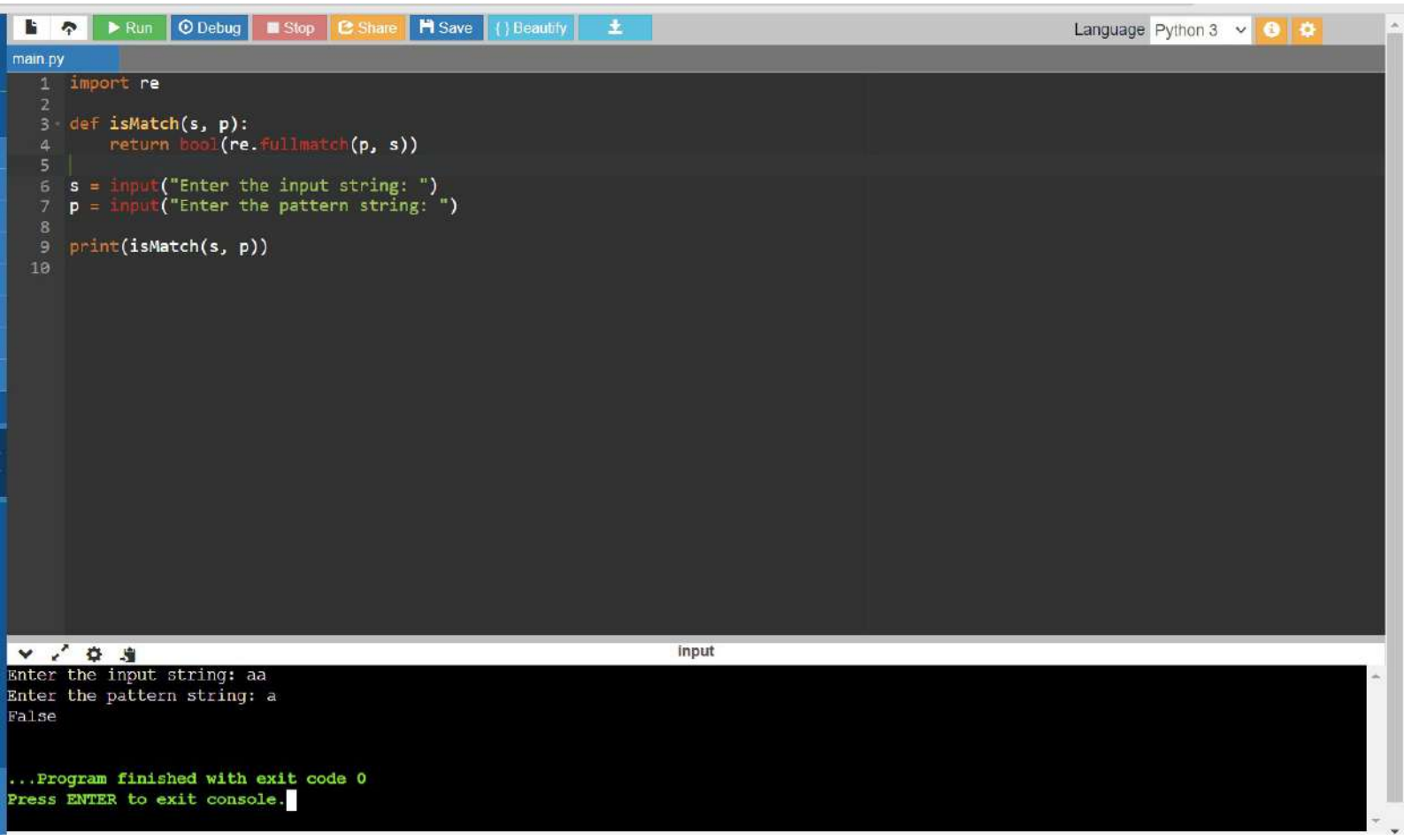
input

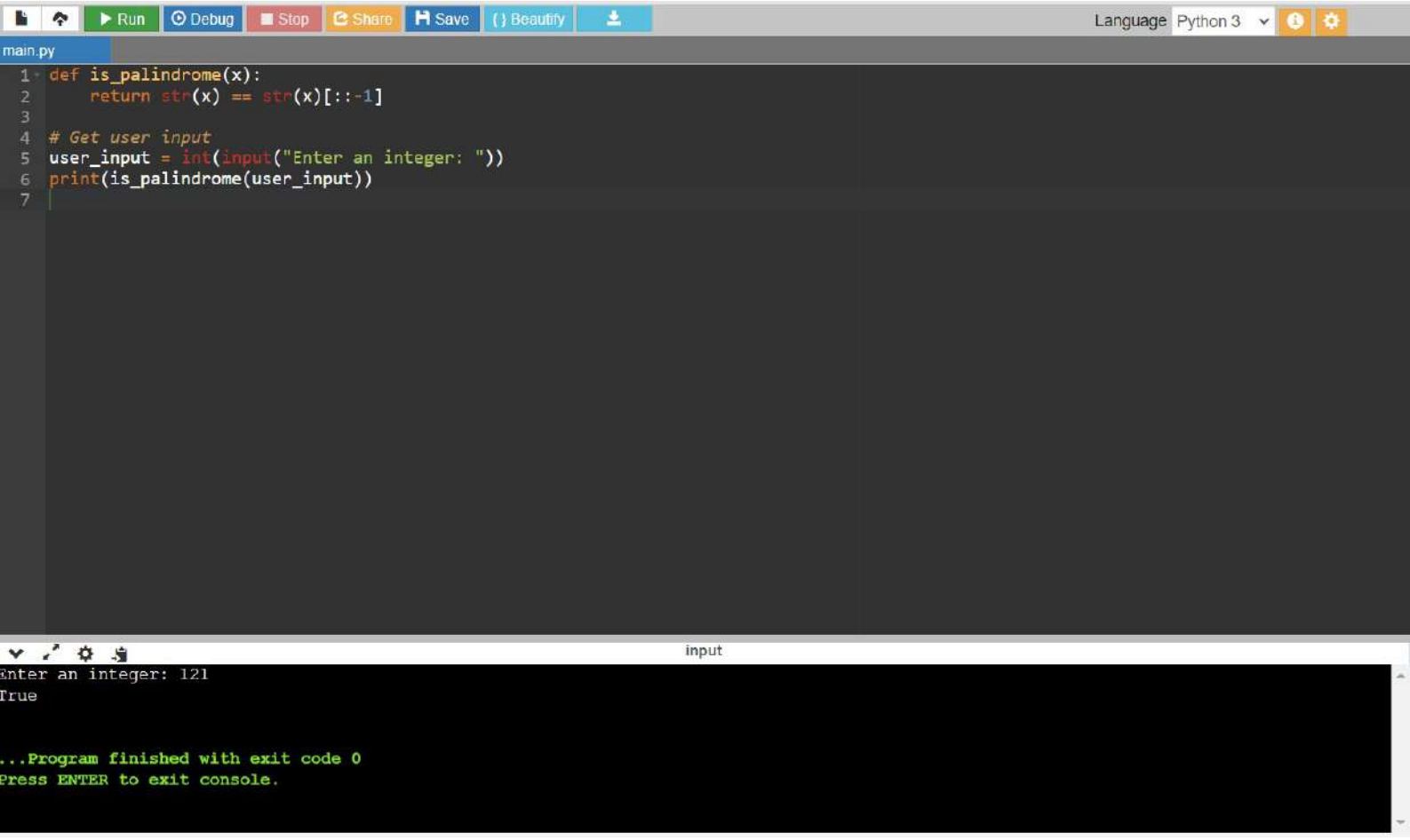
```
7 -> 0 -> 8 -> None


...Program finished with exit code 0
Press ENTER to exit console.
```

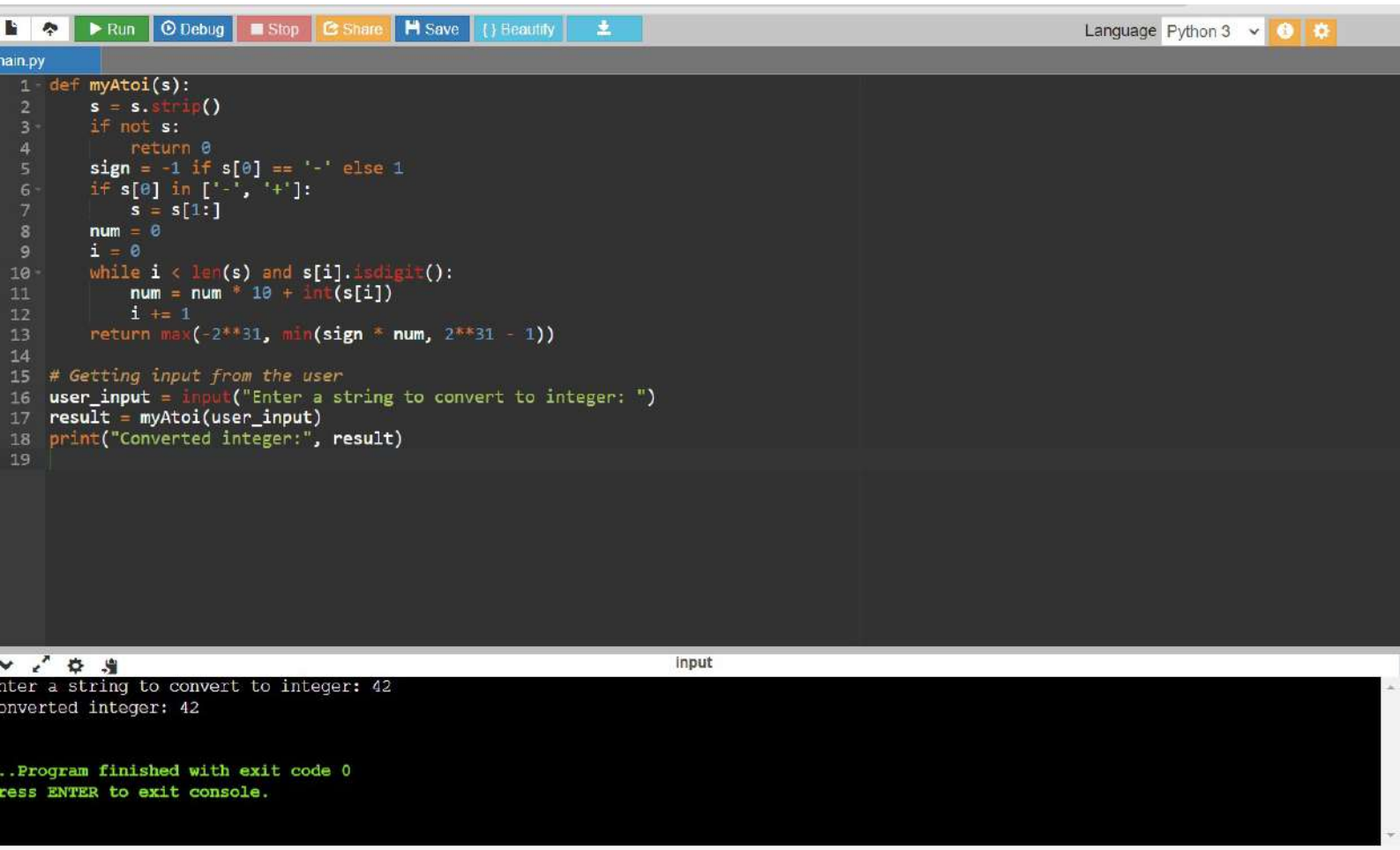
```python
            l1 = l1.next
        if l2:
            l2 = l2.next

    return dummy_head.next


def create_linked_list(lst):
    dummy = ListNode(0)
    current = dummy
    for number in lst:
        current.next = ListNode(number)
        current = current.next
    return dummy.next


def print_linked_list(node):
    while node:
        print(node.val, end=" -> ")
        node = node.next
    print("None")

l1 = create_linked_list([2, 4, 3])
l2 = create_linked_list([5, 6, 4])
result = addTwoNumbers(l1, l2)
print_linked_list(result)
```

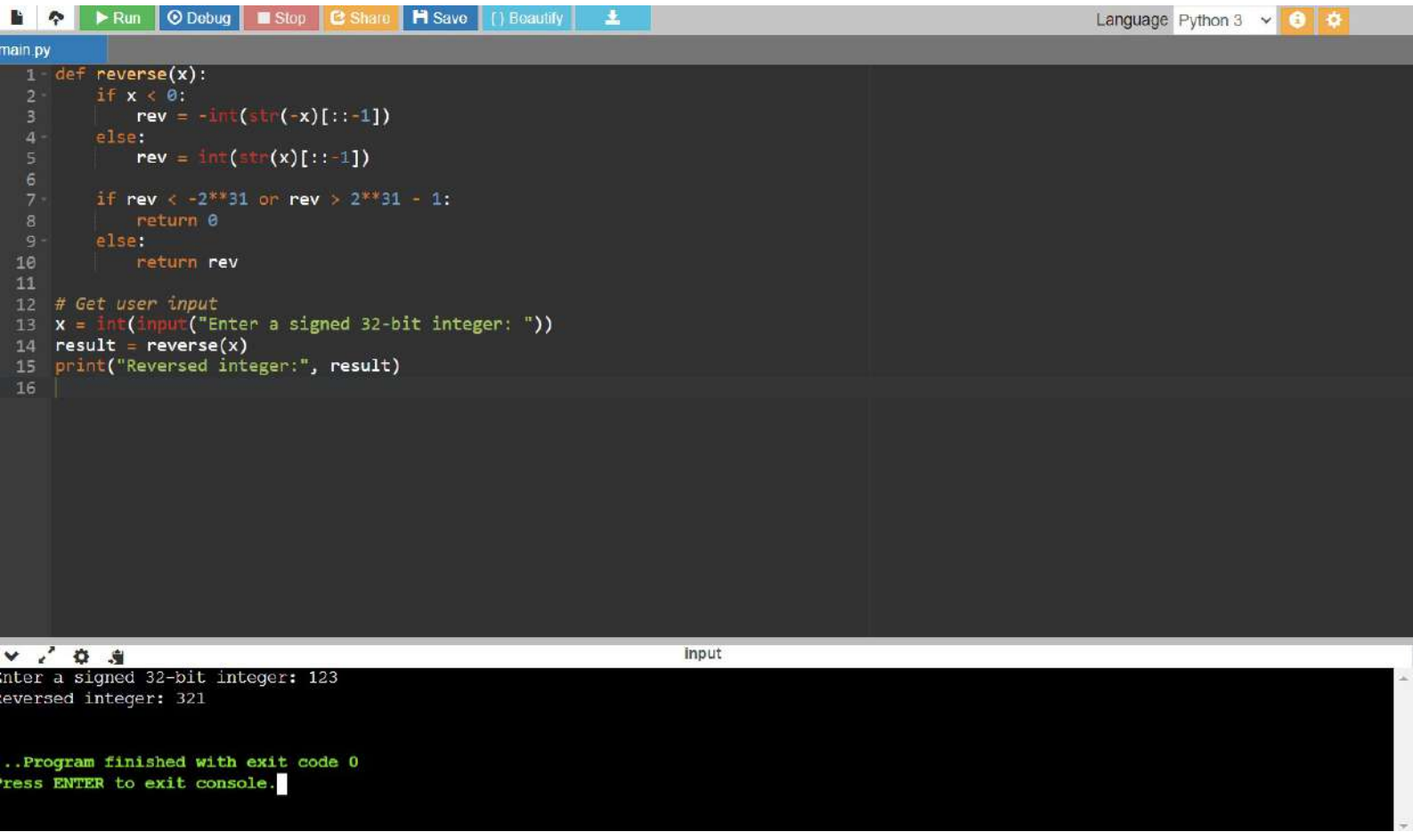
input

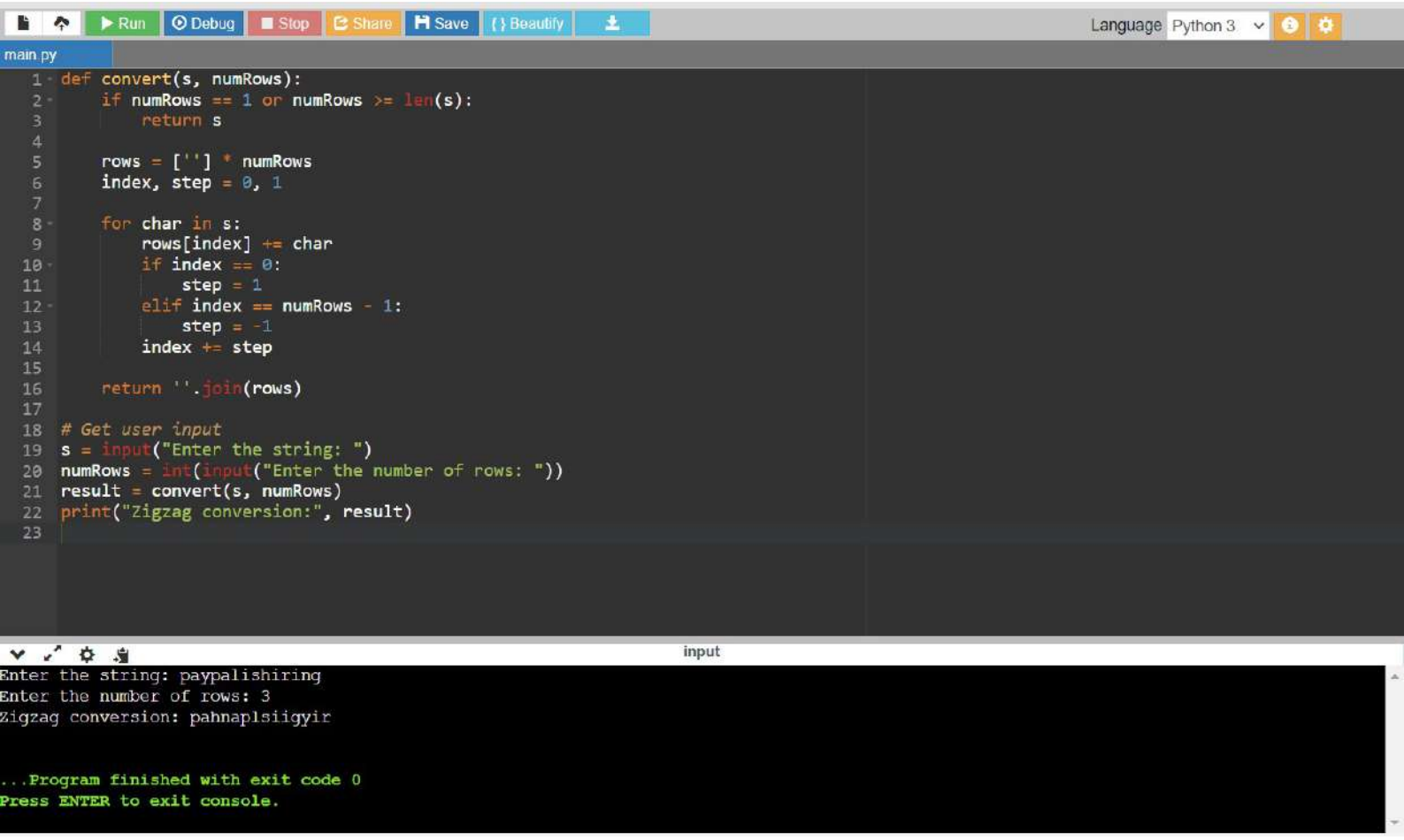
```
7 -> 0 -> 8 -> None


...Program finished with exit code 0
Press ENTER to exit console.
```

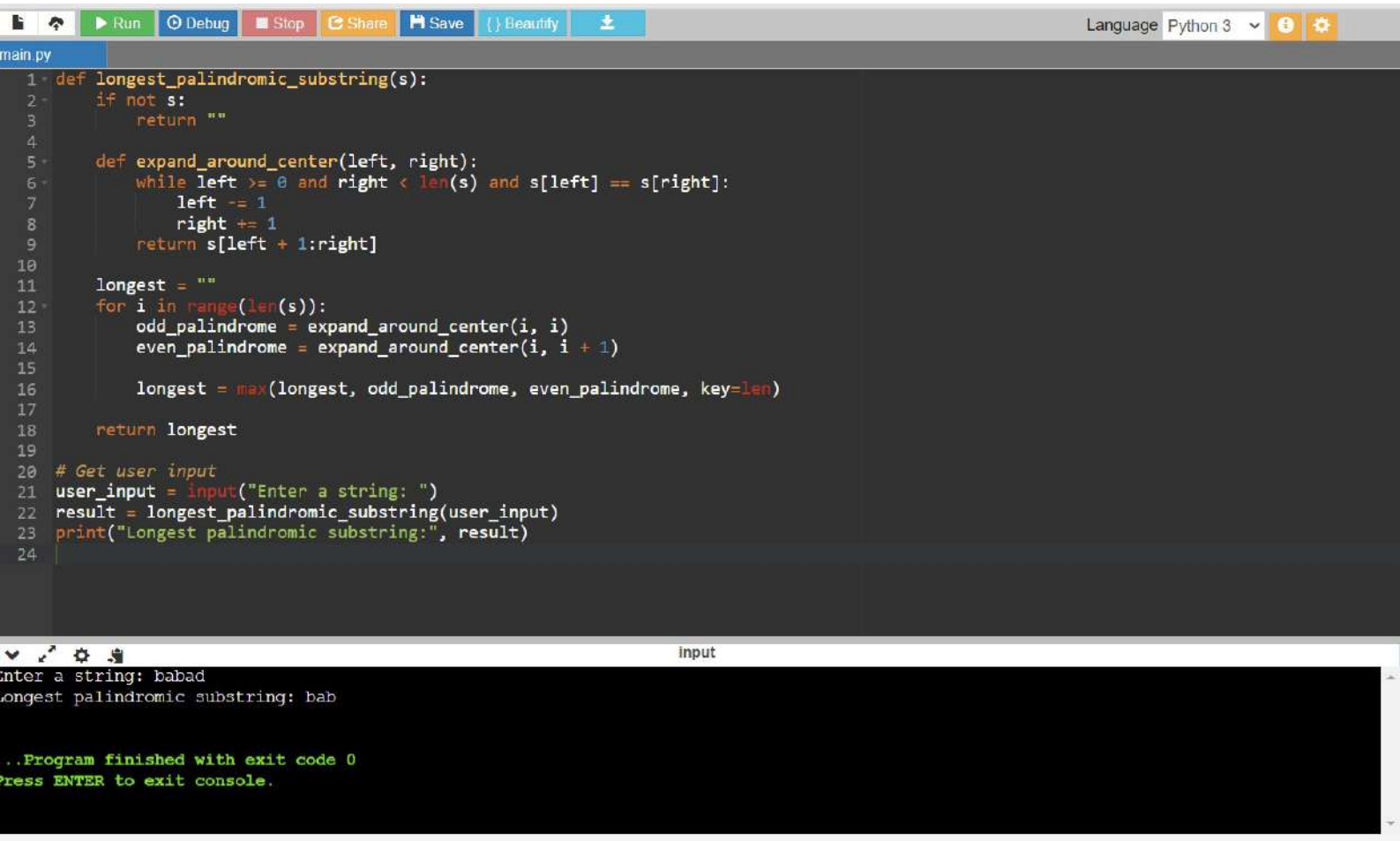

Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
1 import re
2
3 def isMatch(s, p):
4 return bool(re.fullmatch(p, s))
5
6 s = input("Enter the input string: ")
7 p = input("Enter the pattern string: ")
8
9 print(isMatch(s, p))
10
input
Enter the input string: aa
Enter the pattern string: a
False
...Program finished with exit code 0
Press ENTER to exit console.

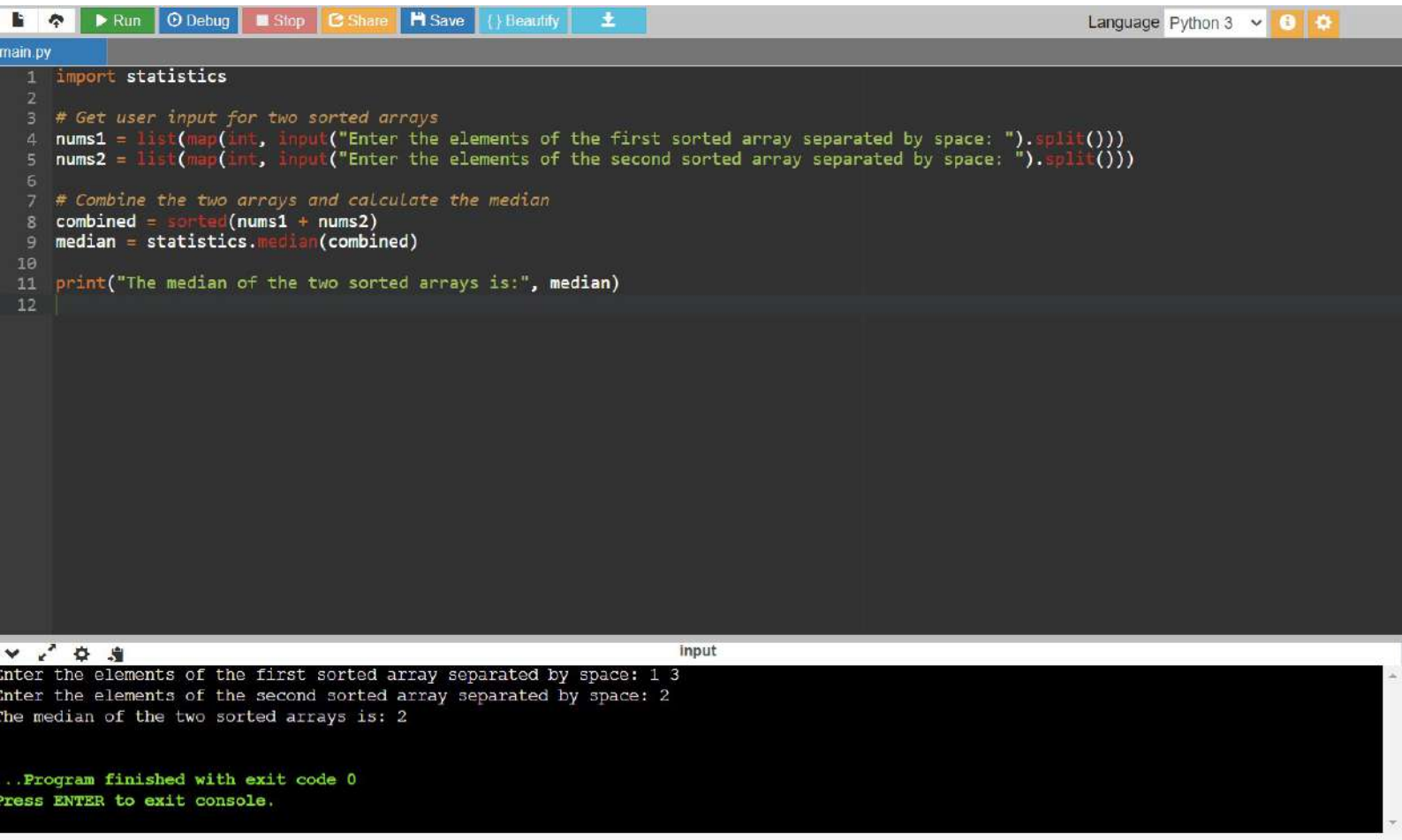

Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
1 def is_palindrome(x):
2 return str(x) == str(x)[::-1]
3
4 # Get user input
5 user_input = int(input("Enter an integer: "))
6 print(is_palindrome(user_input))
7
input
Enter an integer: 121
True
...Program finished with exit code 0
Press ENTER to exit console.


Run
Debug
Stop
Share
Save
{} Beautify
Language
Python 3
main.py
def myAtoi(s):
s = s.strip()
if not s:
return 0
sign = -1 if s[0] == '-' else 1
if s[0] in ['-', '+']:
s = s[1:]
num = 0
i = 0
while i < len(s) and s[i].isdigit():
num = num * 10 + int(s[i])
i += 1
return max(-2**31, min(sign * num, 2**31 - 1))
# Getting input from the user
user_input = input("Enter a string to convert to integer: ")
result = myAtoi(user_input)
print("Converted integer:", result)
input
nter a string to convert to integer: 42
onverted integer: 42
..Program finished with exit code 0
ress ENTER to exit console.


Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
def reverse(x):
if x < 0:
rev = -int(str(-x)[::-1])
else:
rev = int(str(x)[::-1])
if rev < -2**31 or rev > 2**31 - 1:
return 0
else:
return rev
# Get user input
x = int(input("Enter a signed 32-bit integer: "))
result = reverse(x)
print("Reversed integer:", result)
input
Enter a signed 32-bit integer: 123
Reversed integer: 321
...Program finished with exit code 0
Press ENTER to exit console.


Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
def convert(s, numRows):
if numRows == 1 or numRows >= len(s):
return s
rows = [''] * numRows
index, step = 0, 1
for char in s:
rows[index] += char
if index == 0:
step = 1
elif index == numRows - 1:
step = -1
index += step
return ''.join(rows)
# Get user input
s = input("Enter the string: ")
numRows = int(input("Enter the number of rows: "))
result = convert(s, numRows)
print("Zigzag conversion:", result)
input
Enter the string: paypalishiring
Enter the number of rows: 3
Zigzag conversion: pahnaplsiigyir
...Program finished with exit code 0
Press ENTER to exit console.


Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
def longest_palindromic_substring(s):
if not s:
return ""
def expand_around_center(left, right):
while left >= 0 and right < len(s) and s[left] == s[right]:
left -= 1
right += 1
return s[left + 1:right]
longest = ""
for i in range(len(s)):
odd_palindrome = expand_around_center(i, i)
even_palindrome = expand_around_center(i, i + 1)
longest = max(longest, odd_palindrome, even_palindrome, key=len)
return longest
# Get user input
user_input = input("Enter a string: ")
result = longest_palindromic_substring(user_input)
print("Longest palindromic substring:", result)
input
Enter a string: babad
Longest palindromic substring: bab
..Program finished with exit code 0
Press ENTER to exit console.


Run
Debug
Stop
Share
Save
{} Beautify
Language
Python 3
main.py
import statistics
# Get user input for two sorted arrays
nums1 = list(map(int, input("Enter the elements of the first sorted array separated by space: ").split()))
nums2 = list(map(int, input("Enter the elements of the second sorted array separated by space: ").split()))
# Combine the two arrays and calculate the median
combined = sorted(nums1 + nums2)
median = statistics.median(combined)
print("The median of the two sorted arrays is:", median)
input
Enter the elements of the first sorted array separated by space: 1 3
Enter the elements of the second sorted array separated by space: 2
The median of the two sorted arrays is: 2
..Program finished with exit code 0
Press ENTER to exit console.

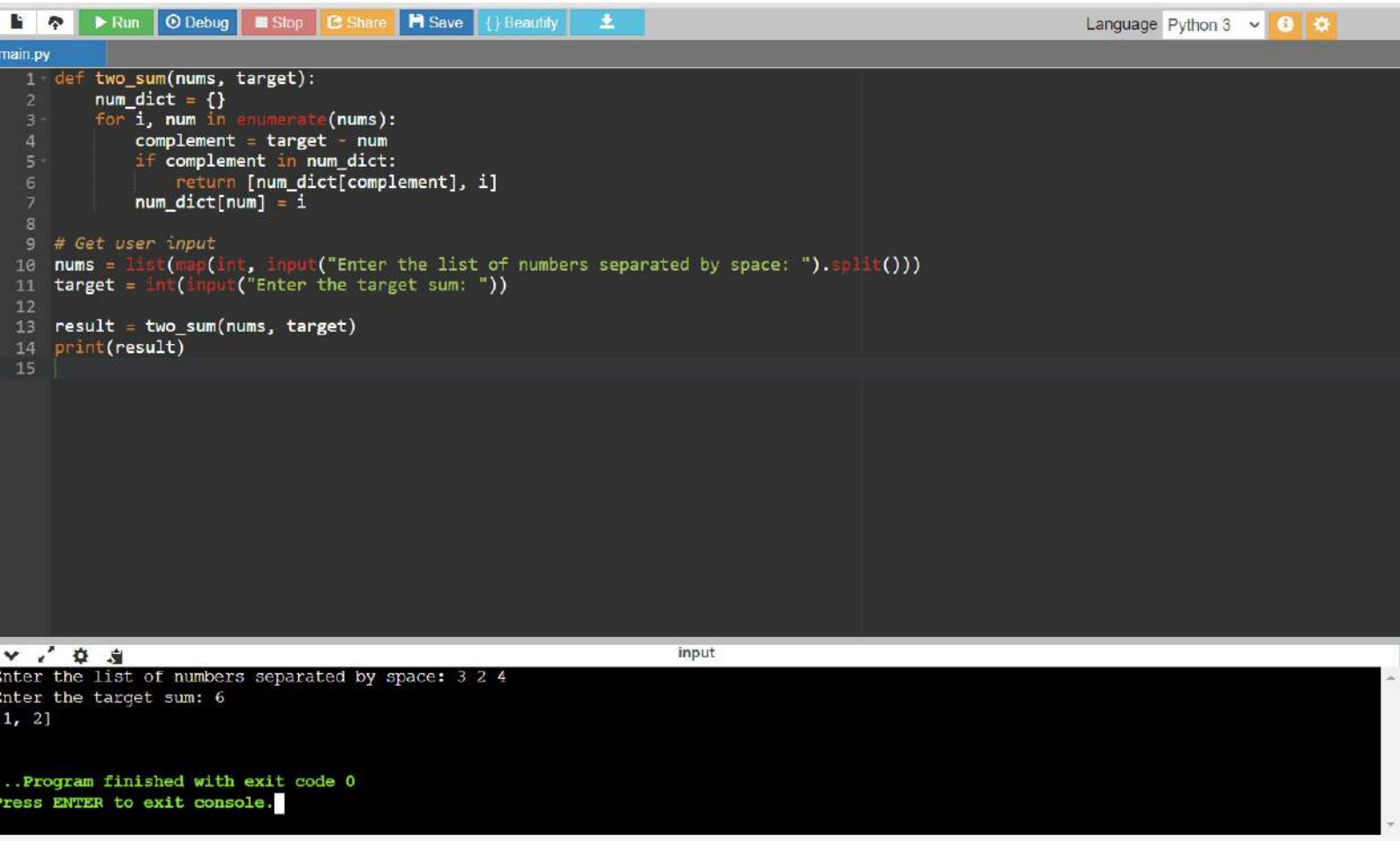
Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
def length_of_longest_substring(s):
    start = maxLength = 0
    used_chars = {}

    for i, char in enumerate(s):
        if char in used_chars and start <= used_chars[char]:
            start = used_chars[char] + 1
        else:
            maxLength = max(maxLength, i - start + 1)

        used_chars[char] = i

    return maxLength

s = input("Enter a string: ")
result = length_of_longest_substring(s)
print("Length of the longest substring without repeating characters:", result)
```

input

```
Enter a string: abcabcbb
Length of the longest substring without repeating characters: 3


...Program finished with exit code 0
Press ENTER to exit console.
```


Run
Debug
Stop
Share
Save
{ } Beautify
Language
Python 3
main.py
def two_sum(nums, target):
num_dict = {}
for i, num in enumerate(nums):
complement = target - num
if complement in num_dict:
return [num_dict[complement], i]
num_dict[num] = i
# Get user input
nums = list(map(int, input("Enter the list of numbers separated by space: ").split()))
target = int(input("Enter the target sum: "))
result = two_sum(nums, target)
print(result)
input
Enter the list of numbers separated by space: 3 2 4
Enter the target sum: 6
1, 2]
..Program finished with exit code 0
ress ENTER to exit console.

Run | Debug | Stop | Share | Save | { } Beautify | Language Python 3

main.py

```python
def isValid(s):
    stack = []
    mapping = {")": "(", "}": "{", "]": "["}
    for char in s:
        if char in mapping:
            top_element = stack.pop() if stack else '#'
            if mapping[char] != top_element:
                return False
        else:
            stack.append(char)
    return not stack

s = "()"
print(isValid(s))

```

input

```
True


...Program finished with exit code 0
Press ENTER to exit console.
```