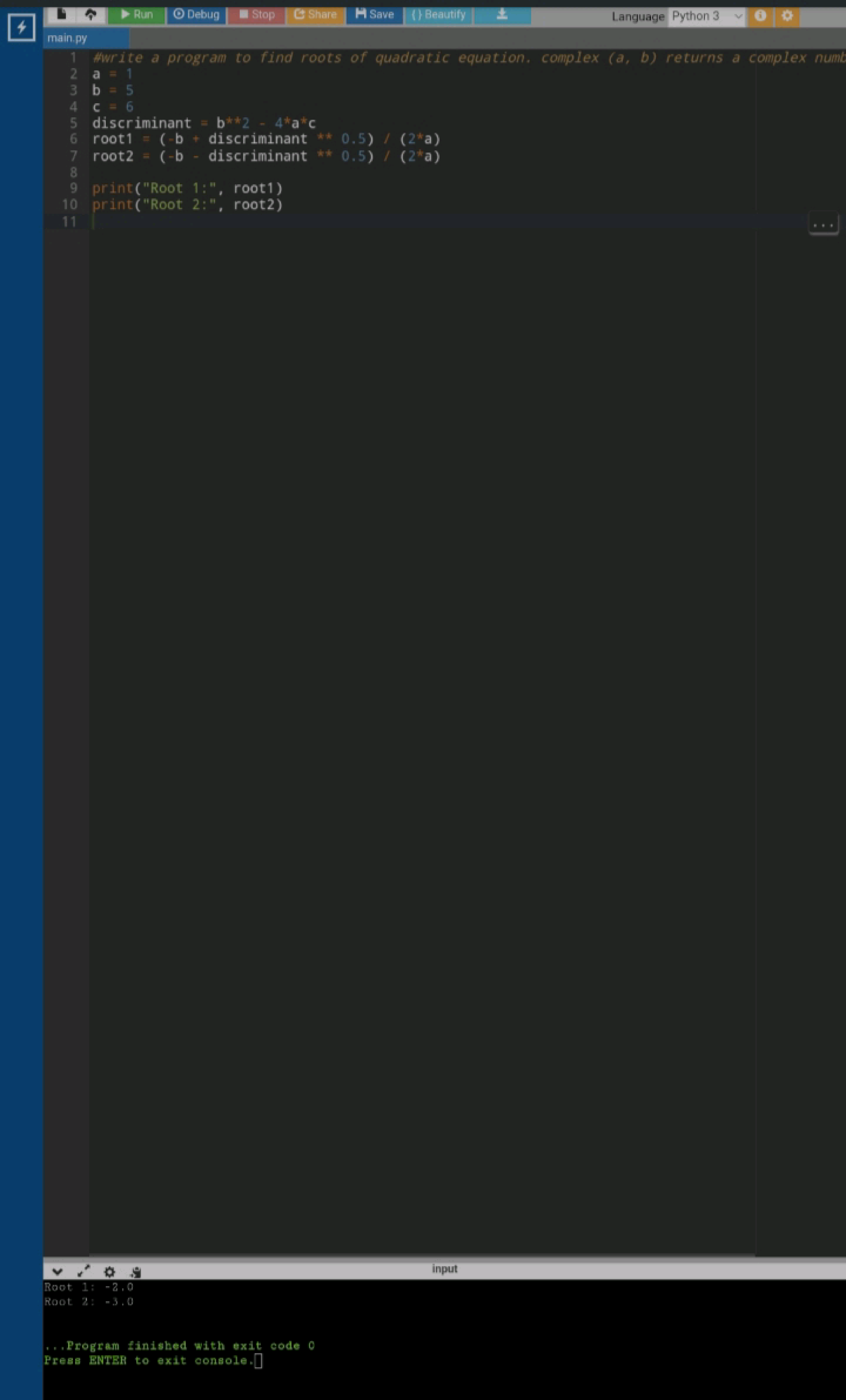
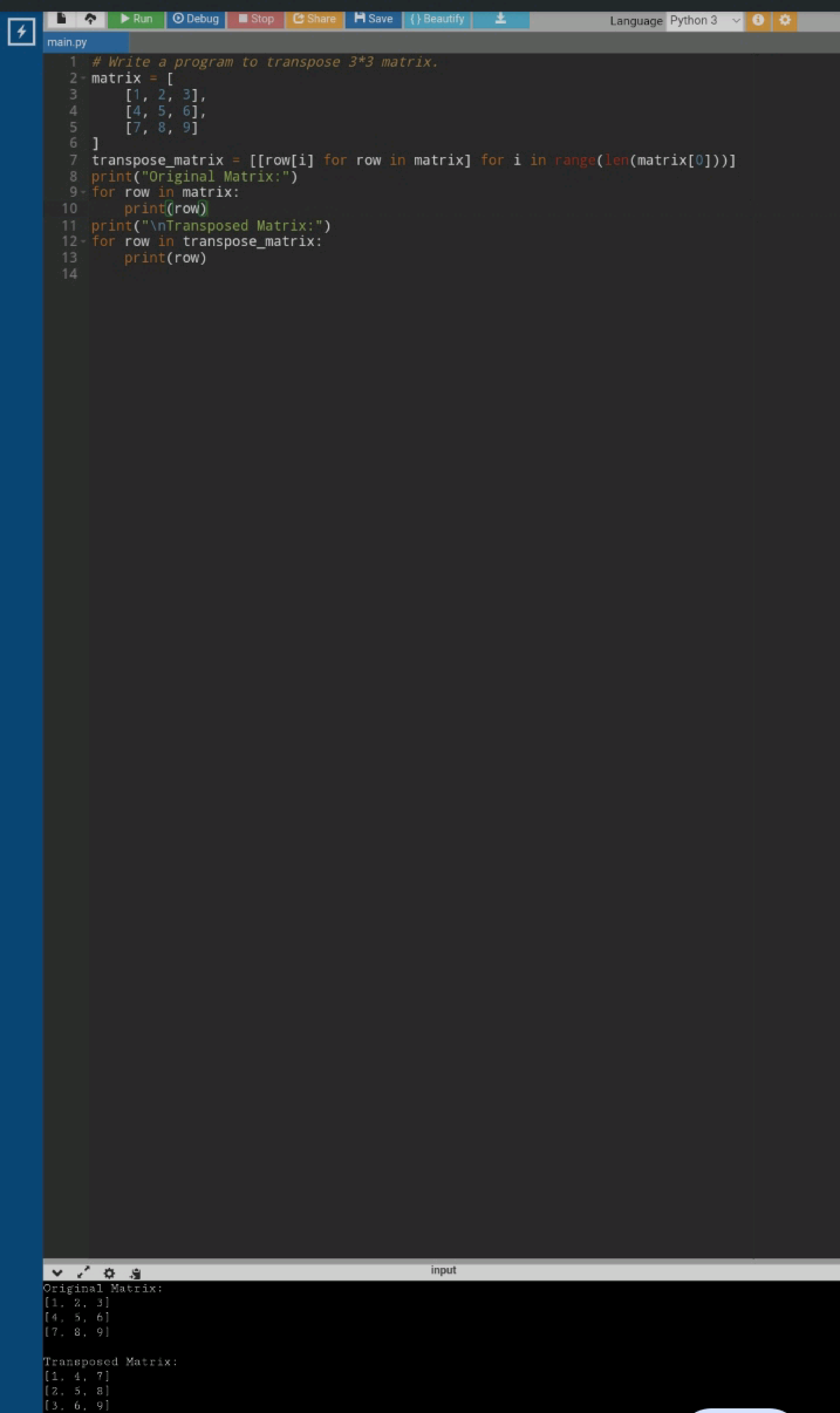


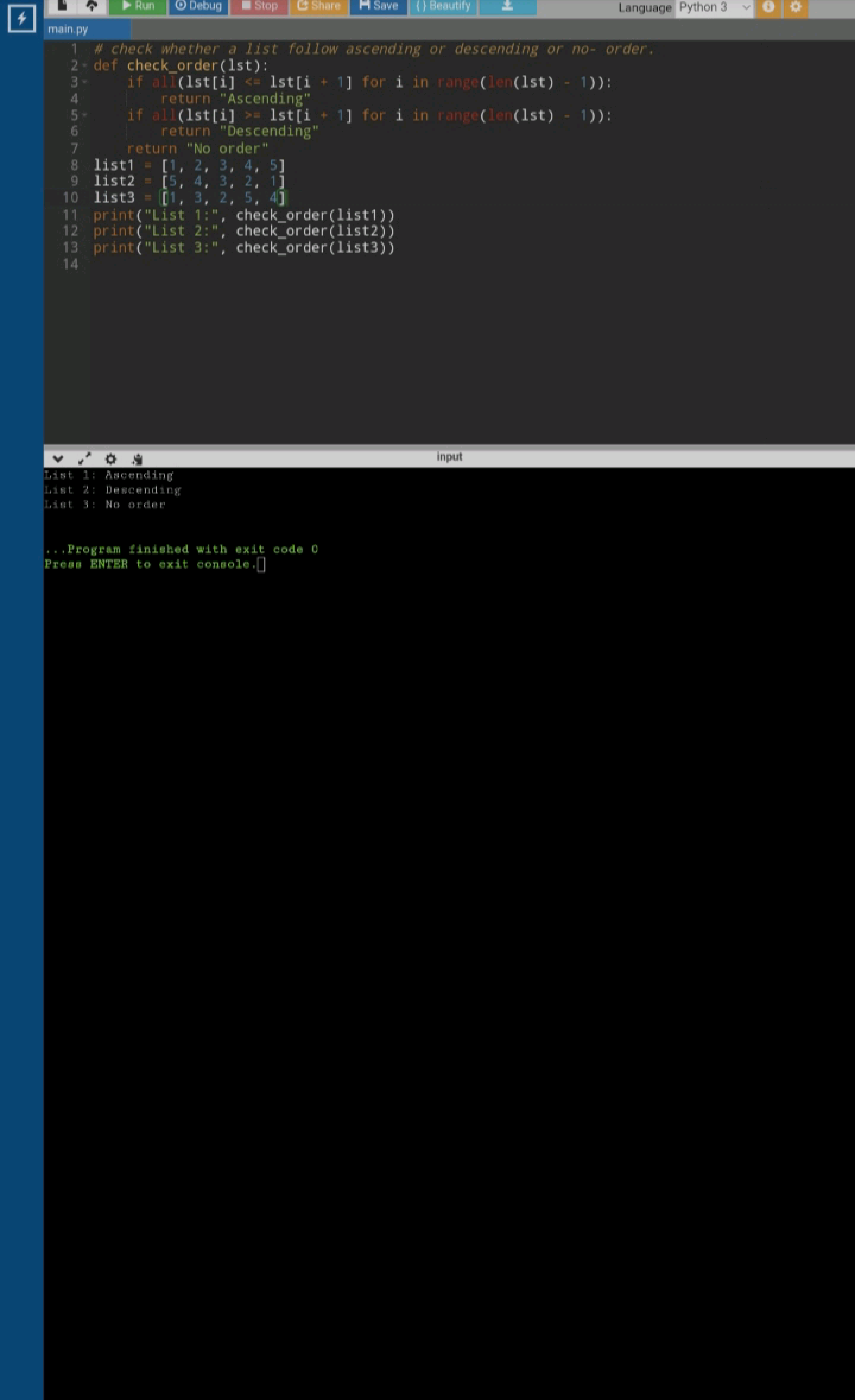
```python
#write a program to find roots of quadratic equation. complex (a, b) returns a complex numb
a = 1
b = 5
c = 6
discriminant = b**2 - 4*a*c
root1 = (-b + discriminant ** 0.5) / (2*a)
root2 = (-b - discriminant ** 0.5) / (2*a)

print("Root 1:", root1)
print("Root 2:", root2)
```

```
Root 1: -2.0
Root 2: -3.0


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



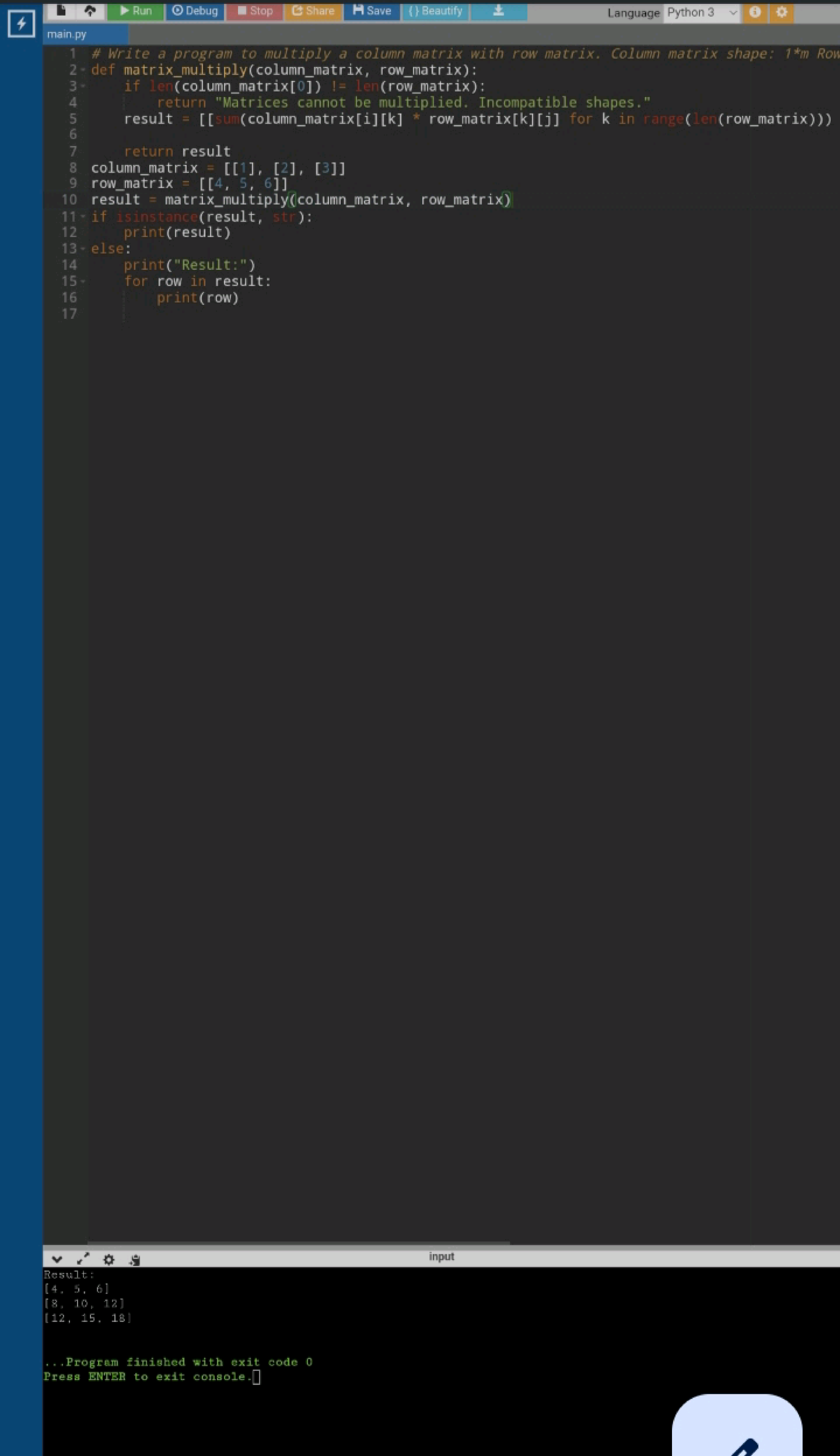
```python
# Write a program to transpose 3*3 matrix.
matrix = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]
transpose_matrix = [[row[i] for row in matrix] for i in range(len(matrix[0]))]
print("Original Matrix:")
for row in matrix:
    print(row)
print("\nTransposed Matrix:")
for row in transpose_matrix:
    print(row)
```

```
Original Matrix:
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]

Transposed Matrix:
[1, 4, 7]
[2, 5, 8]
[3, 6, 9]
```



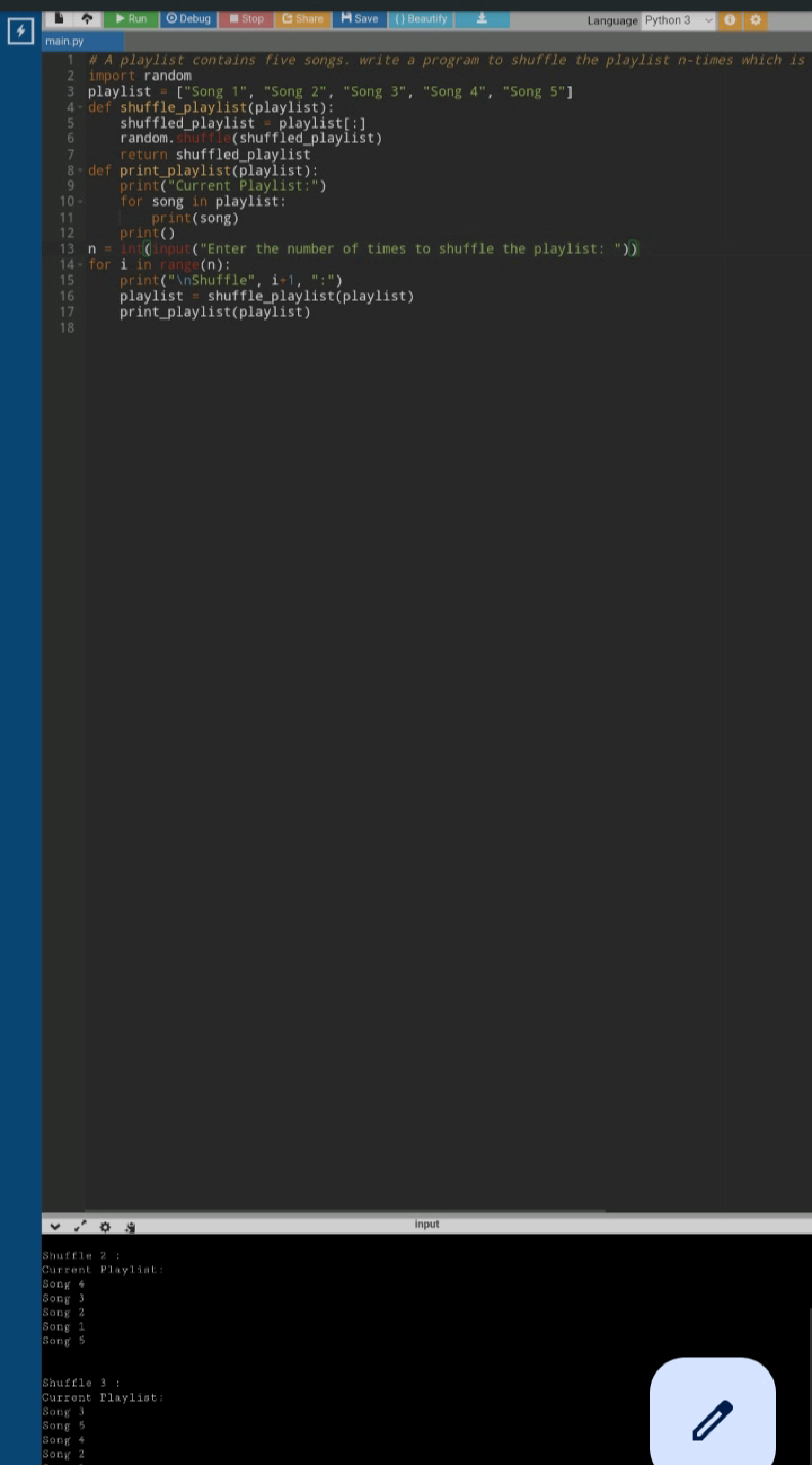
```python
# check whether a list follow ascending or descending or no- order.
def check_order(lst):
    if all(lst[i] <= lst[i + 1] for i in range(len(lst) - 1)):
        return "Ascending"
    if all(lst[i] >= lst[i + 1] for i in range(len(lst) - 1)):
        return "Descending"
    return "No order"
list1 = [1, 2, 3, 4, 5]
list2 = [5, 4, 3, 2, 1]
list3 = [1, 3, 2, 5, 4]
print("List 1:", check_order(list1))
print("List 2:", check_order(list2))
print("List 3:", check_order(list3))
```

```
List 1: Ascending
List 2: Descending
List 3: No order


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



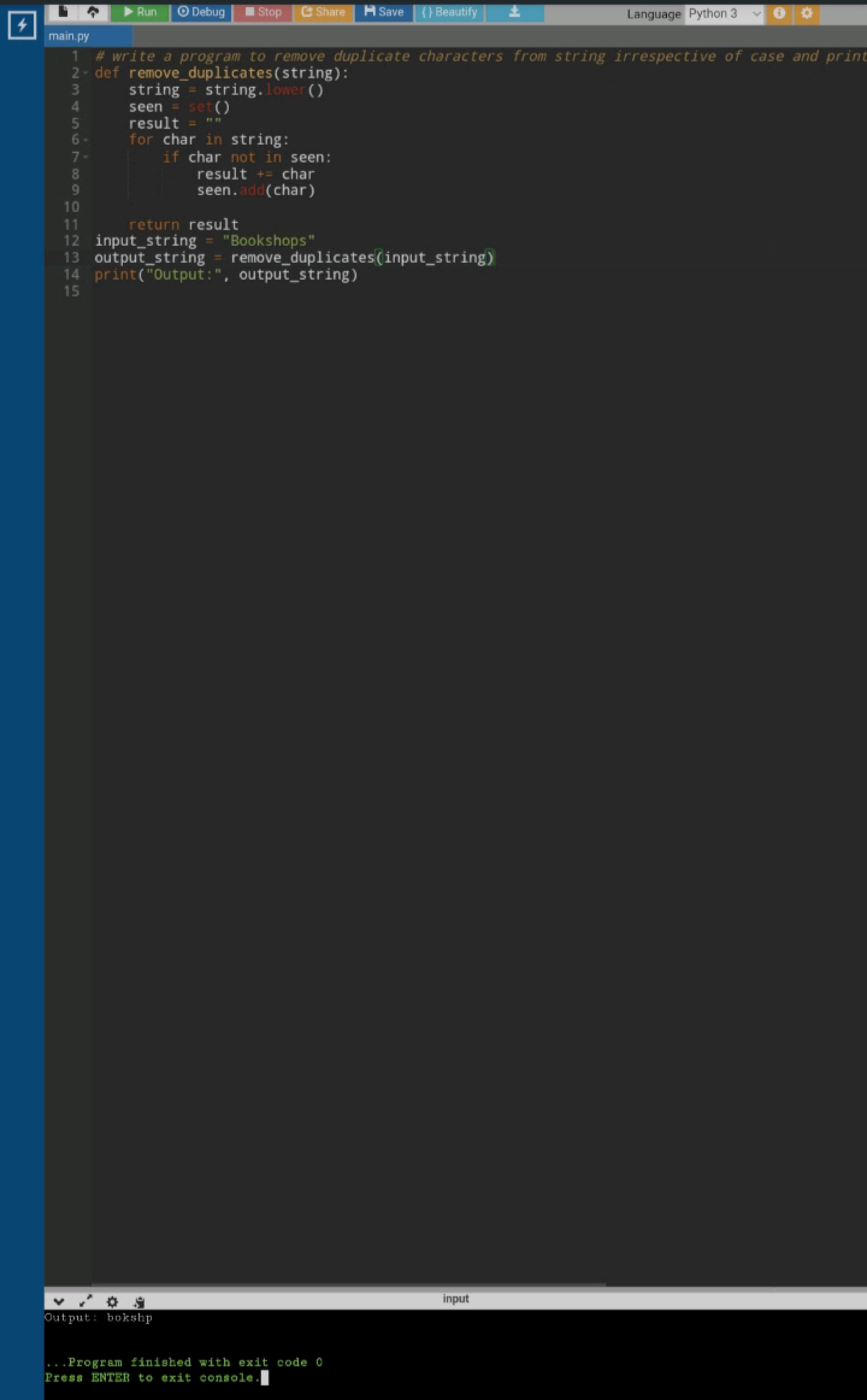
```python
# Write a program to multiply a column matrix with row matrix. Column matrix shape: 1*m Row
def matrix_multiply(column_matrix, row_matrix):
    if len(column_matrix[0]) != len(row_matrix):
        return "Matrices cannot be multiplied. Incompatible shapes."
    result = [[sum(column_matrix[i][k] * row_matrix[k][j] for k in range(len(row_matrix)))

    return result
column_matrix = [[1], [2], [3]]
row_matrix = [[4, 5, 6]]
result = matrix_multiply(column_matrix, row_matrix)
if isinstance(result, str):
    print(result)
else:
    print("Result:")
    for row in result:
        print(row)
```

```
Result:
[4, 5, 6]
[8, 10, 12]
[12, 15, 18]


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



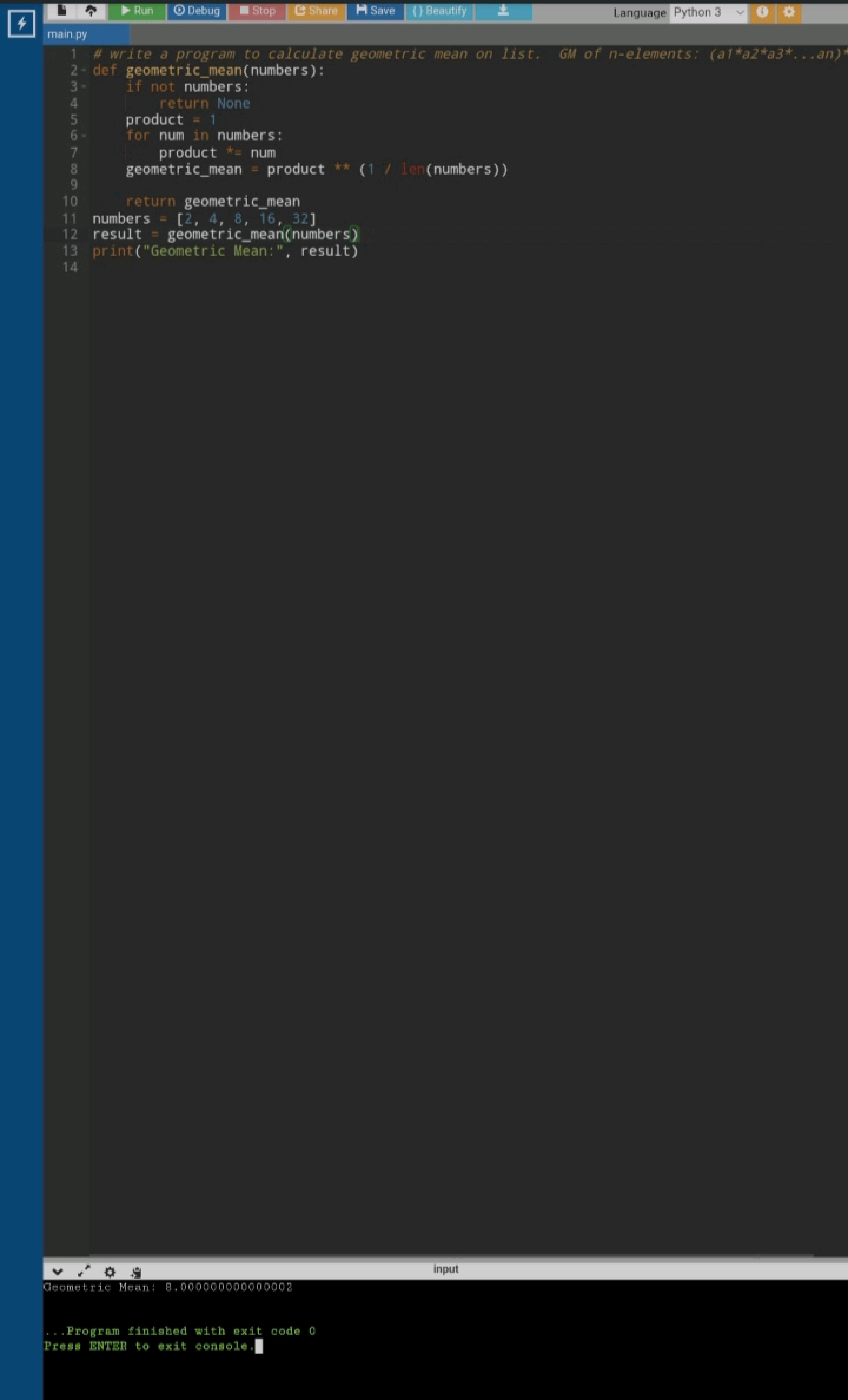
```python
# A playlist contains five songs. write a program to shuffle the playlist n-times which is
import random
playlist = ["Song 1", "Song 2", "Song 3", "Song 4", "Song 5"]
def shuffle_playlist(playlist):
    shuffled_playlist = playlist[:]
    random.shuffle(shuffled_playlist)
    return shuffled_playlist
def print_playlist(playlist):
    print("Current Playlist:")
    for song in playlist:
        print(song)
    print()
n = int(input("Enter the number of times to shuffle the playlist: "))
for i in range(n):
    print("\nShuffle", i+1, ":")
    playlist = shuffle_playlist(playlist)
    print_playlist(playlist)
```

```
Shuffle 2 :
Current Playlist:
Song 4
Song 3
Song 2
Song 1
Song 5


Shuffle 3 :
Current Playlist:
Song 3
Song 5
Song 4
Song 2
```



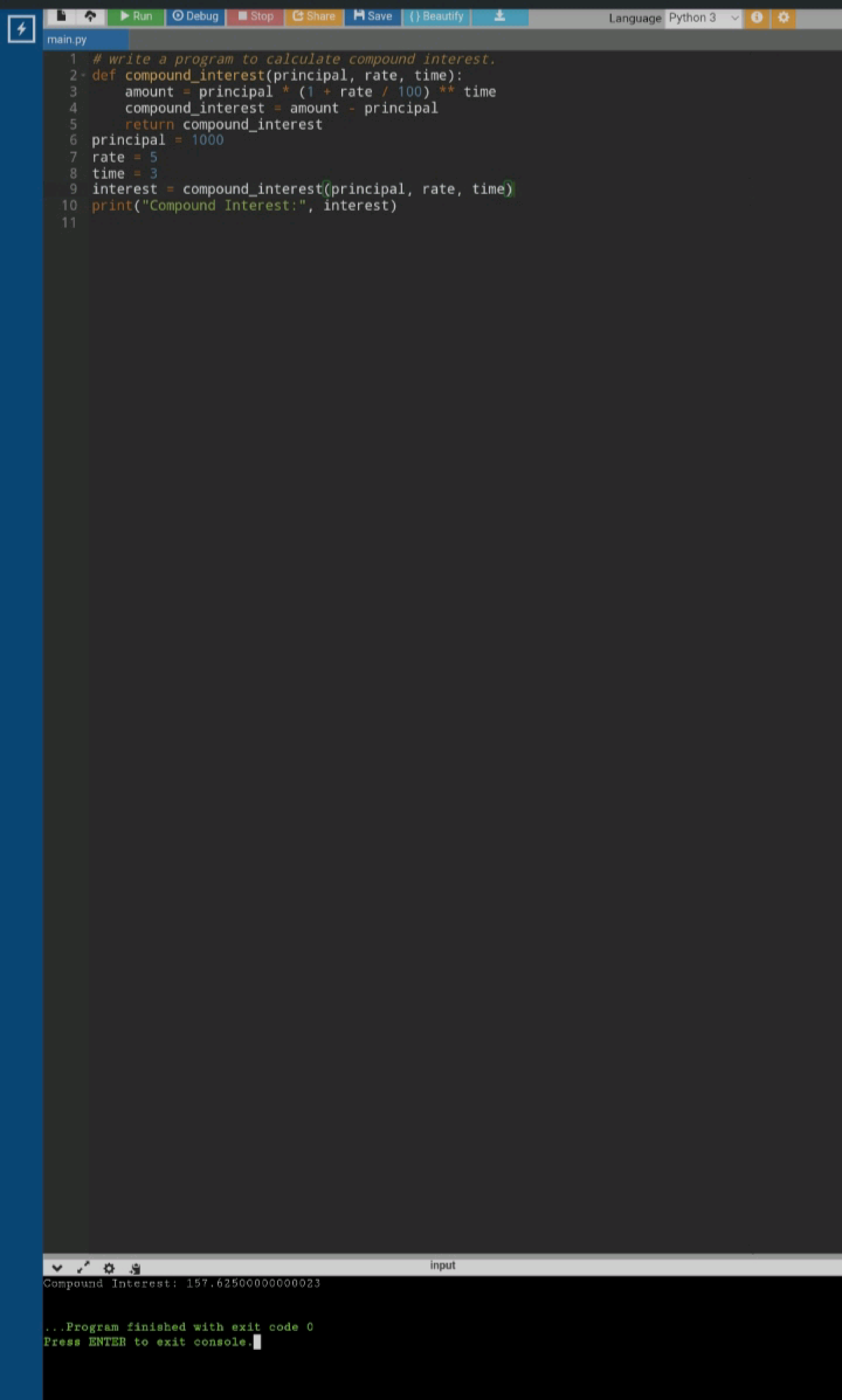
```python
# write a program to remove duplicate characters from string irrespective of case and print
def remove_duplicates(string):
    string = string.lower()
    seen = set()
    result = ""
    for char in string:
        if char not in seen:
            result += char
            seen.add(char)

    return result
input_string = "Bookshops"
output_string = remove_duplicates(input_string)
print("Output:", output_string)
```

```
Output: bokshp


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



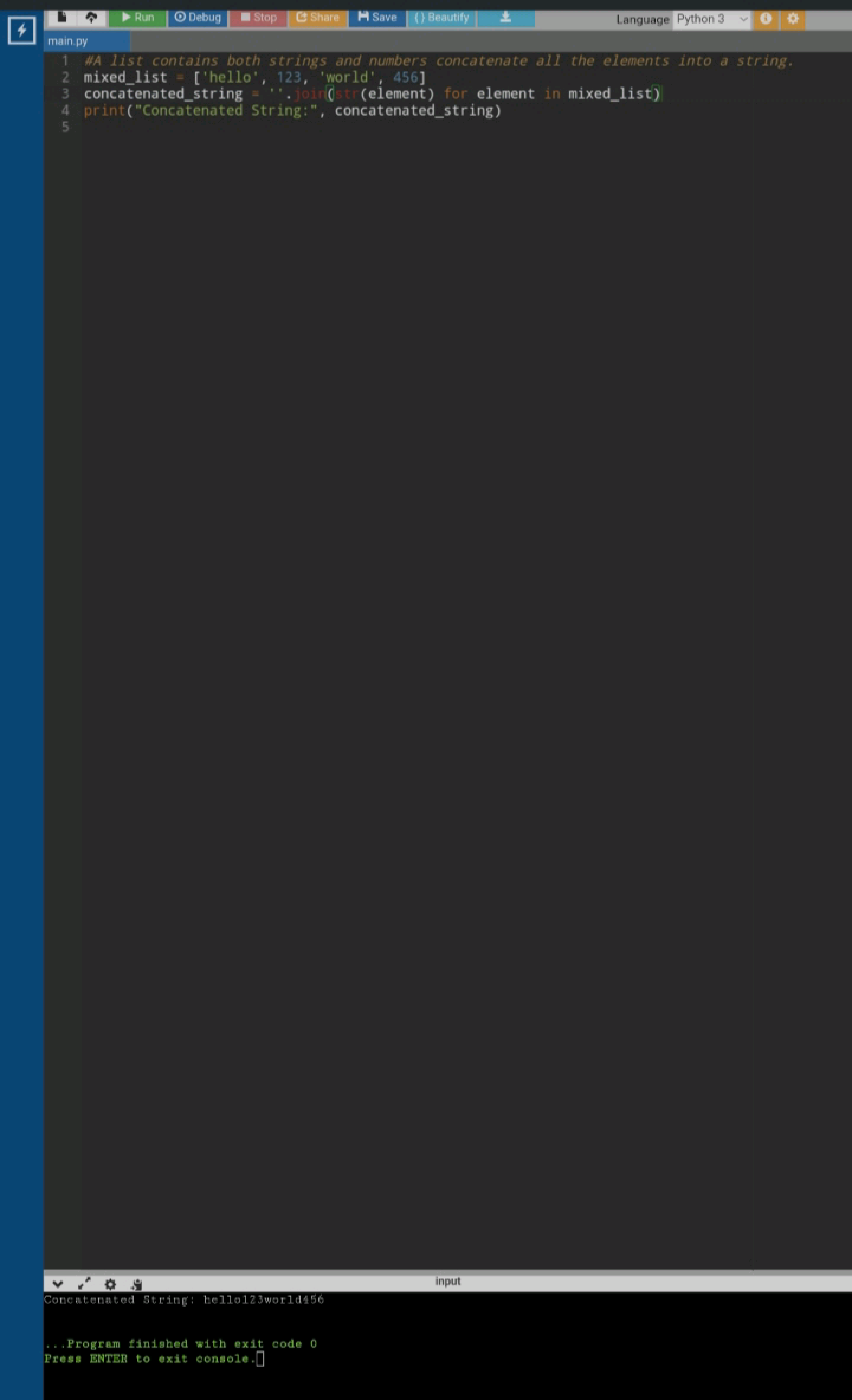
```python
# write a program to calculate geometric mean on list.  GM of n-elements: (a1*a2*a3*...an)*
def geometric_mean(numbers):
    if not numbers:
        return None
    product = 1
    for num in numbers:
        product *= num
    geometric_mean = product ** (1 / len(numbers))

    return geometric_mean
numbers = [2, 4, 8, 16, 32]
result = geometric_mean(numbers)
print("Geometric Mean:", result)
```

```
Geometric Mean: 8.000000000000002


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



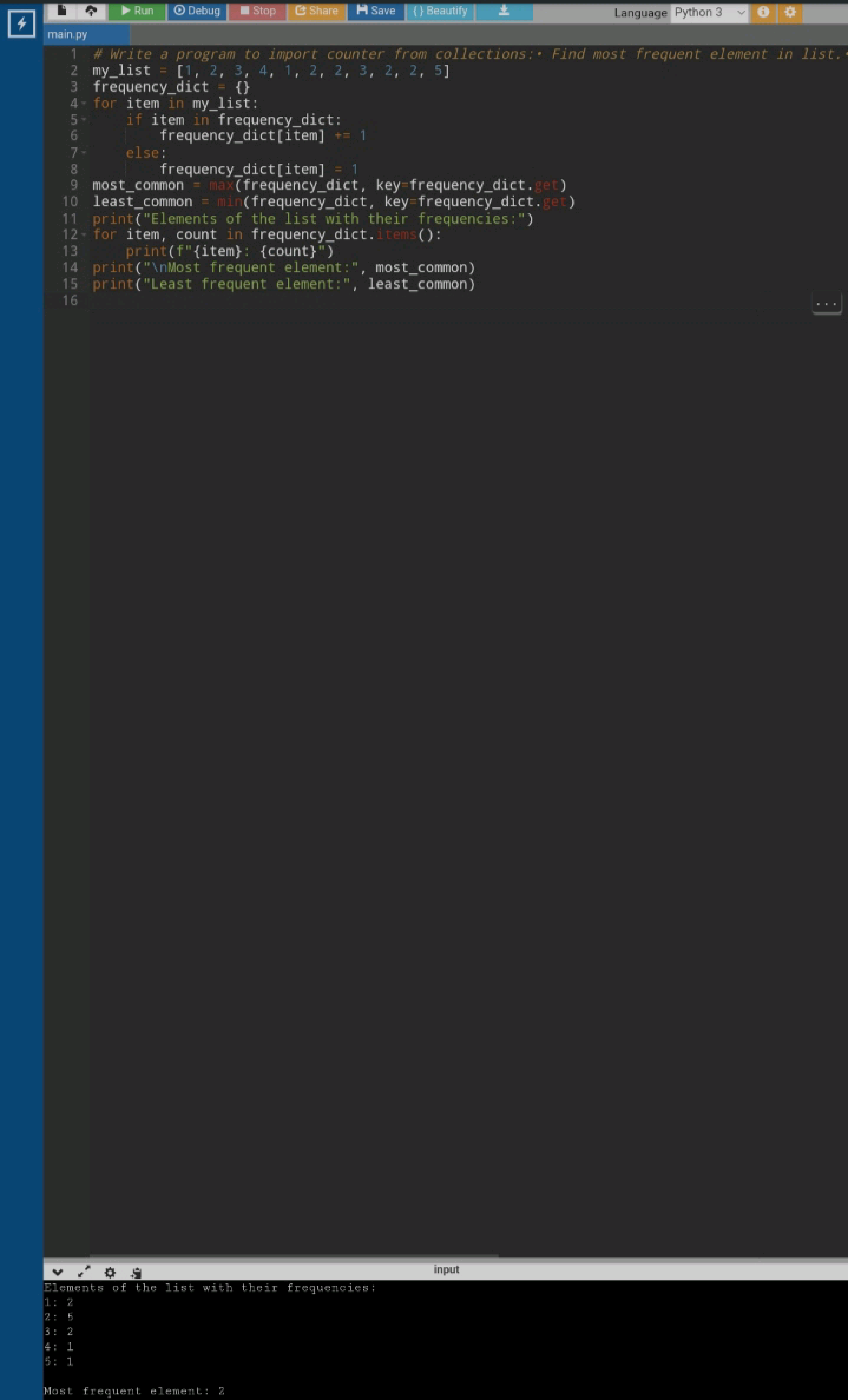
```python
# write a program to calculate compound interest.
def compound_interest(principal, rate, time):
    amount = principal * (1 + rate / 100) ** time
    compound_interest = amount - principal
    return compound_interest
principal = 1000
rate = 5
time = 3
interest = compound_interest(principal, rate, time)
print("Compound Interest:", interest)
```

```
Compound Interest: 157.62500000000023


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



```python
#A list contains both strings and numbers concatenate all the elements into a string.
mixed_list = ['hello', 123, 'world', 456]
concatenated_string = ''.join(str(element) for element in mixed_list)
print("Concatenated String:", concatenated_string)
```

```
Concatenated String: hello123world456


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



```python
# Write a program to import counter from collections:• Find most frequent element in list.•
my_list = [1, 2, 3, 4, 1, 2, 2, 3, 2, 2, 5]
frequency_dict = {}
for item in my_list:
    if item in frequency_dict:
        frequency_dict[item] += 1
    else:
        frequency_dict[item] = 1
most_common = max(frequency_dict, key=frequency_dict.get)
least_common = min(frequency_dict, key=frequency_dict.get)
print("Elements of the list with their frequencies:")
for item, count in frequency_dict.items():
    print(f"{item}: {count}")
print("\nMost frequent element:", most_common)
print("Least frequent element:", least_common)
```

```
Elements of the list with their frequencies:
1: 2
2: 5
3: 2
4: 1
5: 1

Most frequent element: 2
```