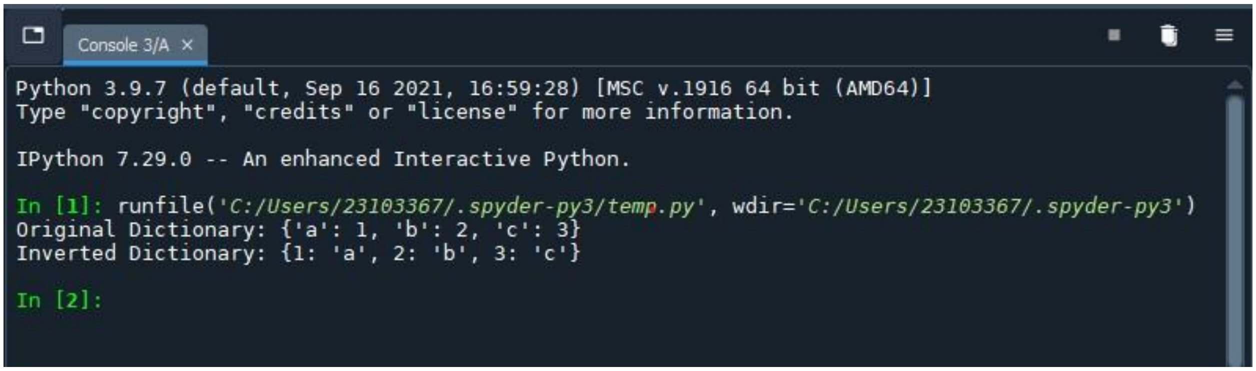
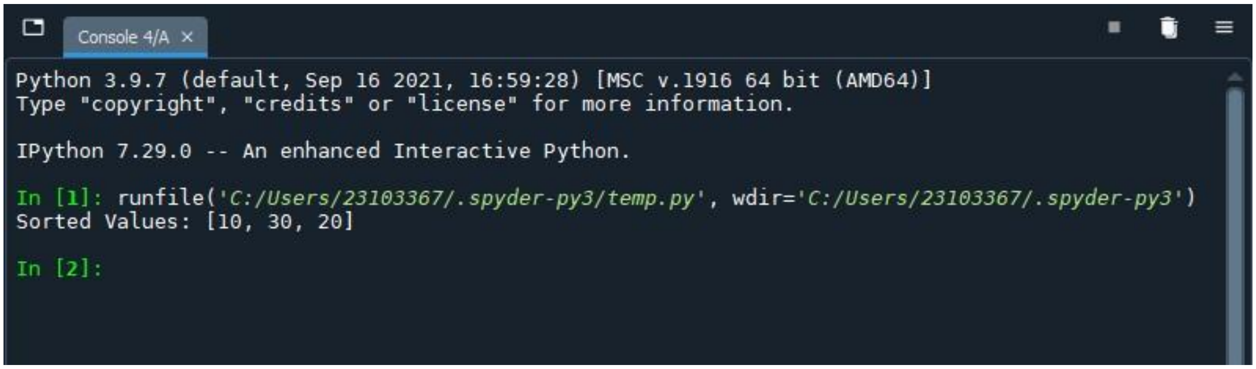
Assignment-03

Question1: def
```
invertdict(d):
    return {v: k for k, v in d.items()}


original_dict = {'a': 1, 'b': 2, 'c': 3}
inverted_dict = invertdict(original_dict)
print("Original Dictionary:", original_dict)
print("Inverted Dictionary:", inverted_dict)
```



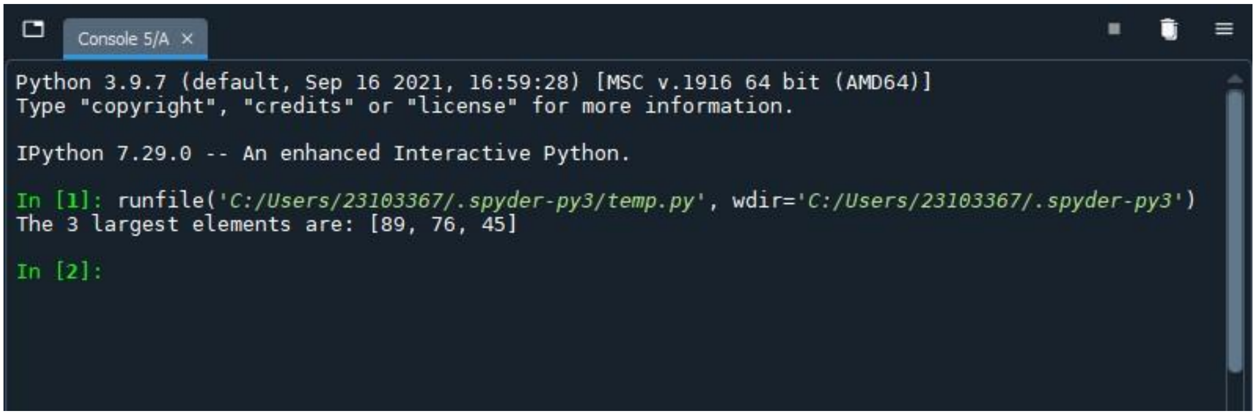
```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
Original Dictionary: {'a': 1, 'b': 2, 'c': 3}
Inverted Dictionary: {1: 'a', 2: 'b', 3: 'c'}

In [2]:
```

Question2:
```
def valuesort(d):
    return [d[k] for k in sorted(d.keys())]

sample_dict = {'b': 30, 'a': 10, 'c': 20}
sorted_values = valuesort(sample_dict)
print("Sorted Values:", sorted_values)
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
Sorted Values: [10, 30, 20]

In [2]:
```

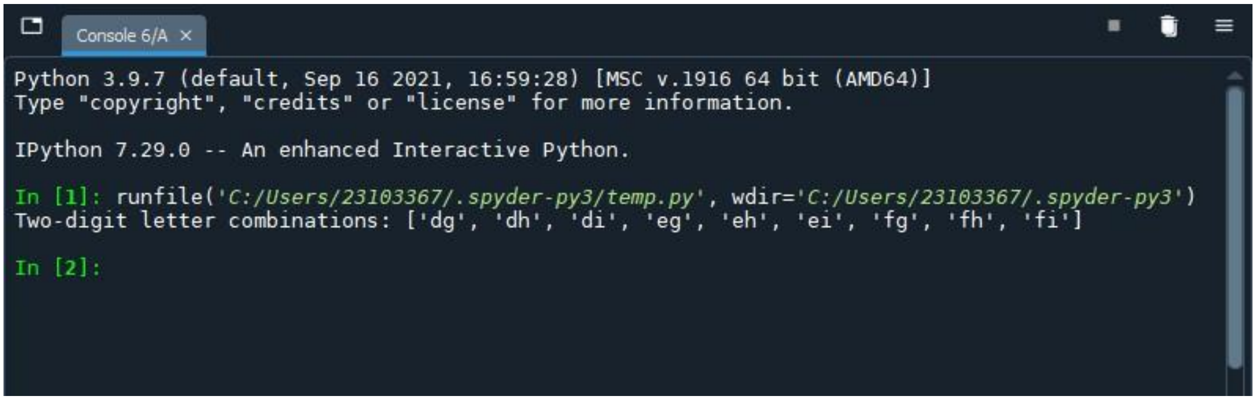
Question3:
```
def n_largest_elements(lst, n):
    return sorted(lst, reverse=True)[:n] numbers = [10,
45, 23, 76, 89, 12, 5] n = 3 largest_elements =
n_largest_elements(numbers, n) print(f"The {n} largest
elements are:", largest_elements)
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
The 3 largest elements are: [89, 76, 45]

In [2]:
```

Question4:

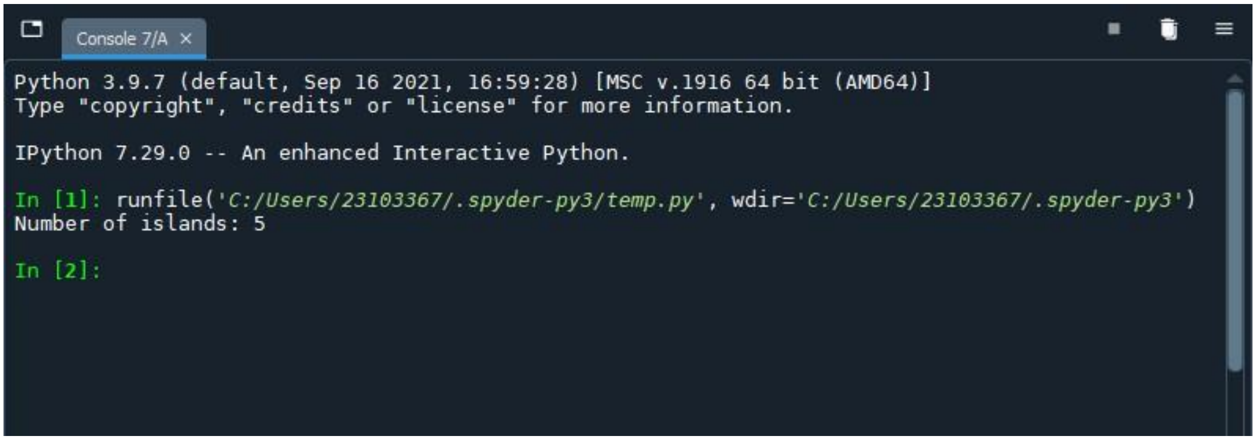
```
from itertools import product

string_maps = {
   "1": "abc",
   "2": "def",
   "3": "ghi",
   "4": "jkl",
   "5": "mno",
"6": "pqrs",
   "7": "tuv",
   "8": "wxy",
   "9": "z"
}

def two_digit_combinations(digits):
if len(digits) < 2:
      return []

   combinations = []     for i in
range(len(digits) - 1):
      first_letters = string_maps.get(digits[i], "")
second_letters = string_maps.get(digits[i + 1], "")         pairs =
[''.join(p) for p in product(first_letters, second_letters)]
combinations.extend(pairs)
   return combinations

input_digits = "23" output =
two_digit_combinations(input_digits)
print("Two-digit letter combinations:", output)
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
Two-digit letter combinations: ['dg', 'dh', 'di', 'eg', 'eh', 'ei', 'fg', 'fh', 'fi']

In [2]:
```

Question5:

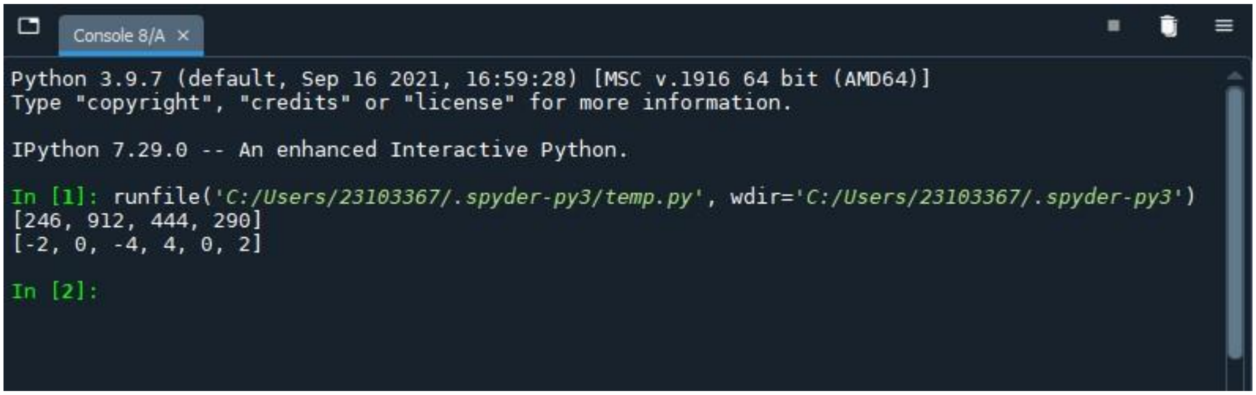
```
def num_islands(grid):     def dfs(i, j):         if i < 0 or j <
0 or i >= 10 or j >= 10 or grid[i][j] != 1:
            return
        grid[i][j] = -1  # Mark as visited
for dx, dy in [(-1,0), (1,0), (0,-1), (0,1)]:
            dfs(i + dx, j + dy)

    count = 0     for i in
range(10):         for j in
range(10):             if
grid[i][j] == 1:
dfs(i, j)
count += 1     return
count

# Input grid (10x10) grid
= [
    [1,1,0,0,0,0,0,1,1,1],
    [1,0,0,0,0,0,0,1,1,1],
    [0,0,0,0,0,0,0,1,1,1],
    [0,0,1,0,0,0,1,0,0,0],
    [0,0,0,0,0,1,1,1,0,0],
    [0,0,0,0,1,1,1,1,1,0],
    [0,0,0,1,1,1,1,1,1,1],
    [1,0,0,0,1,1,1,1,1,0],
    [1,1,0,0,0,1,1,1,0,0],
    [1,1,1,0,0,0,1,0,0,0]
]

island_count = num_islands(grid)
print("Number of islands:", island_count)
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
Number of islands: 5

In [2]:
```

Question6:

```
def double_input_string(input_str):
    return [int(num.strip()) * 2 for num in input_str.split(',')]

input1 = "123, 456, 222, 145" input2
= "-1, 0, -2, 2, 0, 1"

print(double_input_string(input1))  # Output: [246, 912, 444, 290]
print(double_input_string(input2))  # Output: [-2, 0, -4, 4, 0, 2]
```

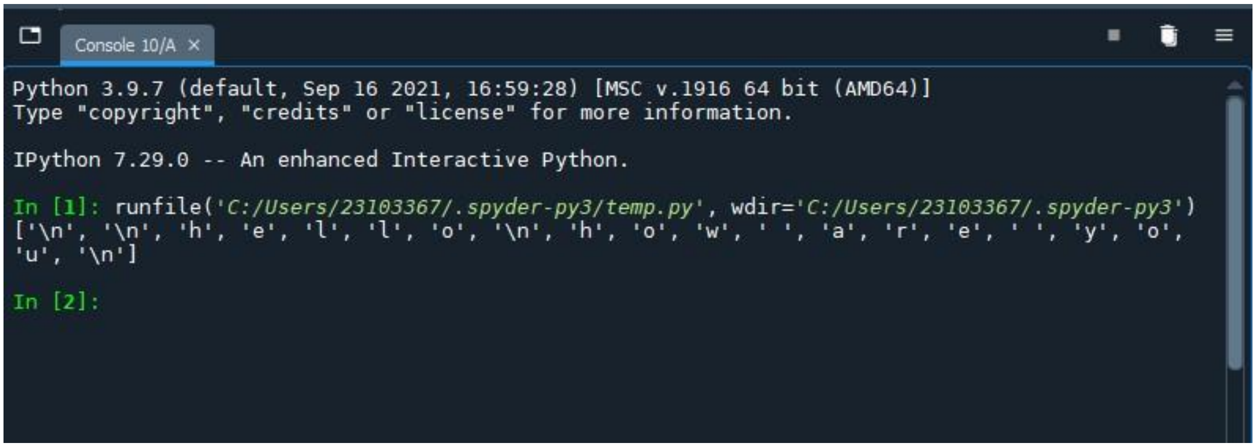


```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
[246, 912, 444, 290]
[-2, 0, -4, 4, 0, 2]

In [2]:
```

Question7:

```
def extract_characters_from_files(filenames):
   all_chars = []      for filename in
filenames:          try:              with
open(filename, 'r') as file:
contents = file.read()
all_chars.extend(list(contents))
except FileNotFoundError:
        print(f"File not found: {filename}")
return all_chars
```

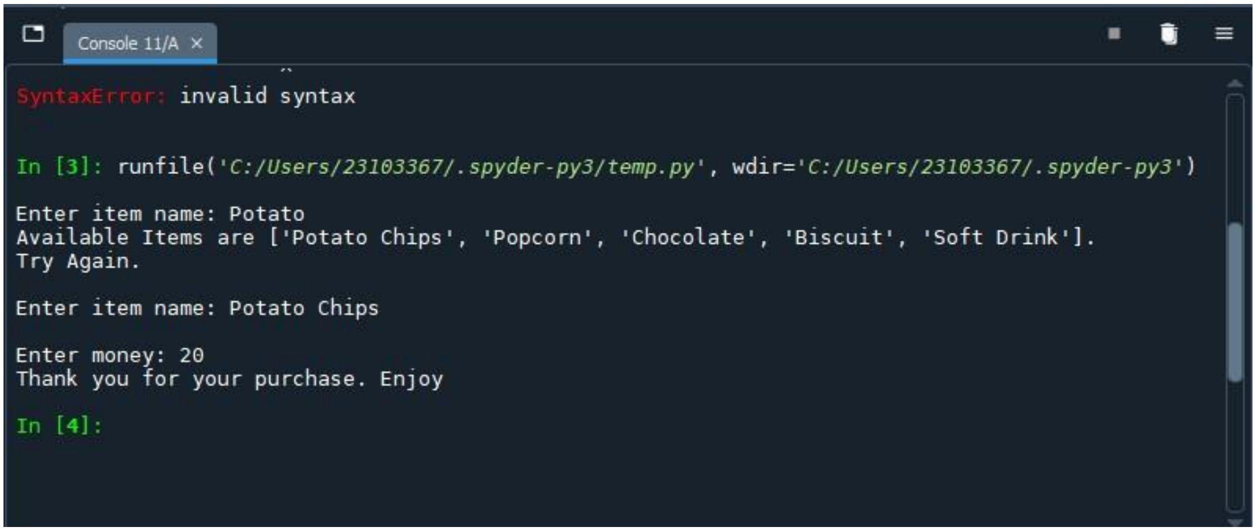
```
file_list = ['abc.txt', 'abcd.txt']   characters =
extract_characters_from_files(file_list)
print(characters)
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
['\n', '\n', 'h', 'e', 'l', 'l', 'o', '\n', 'h', 'o', 'w', ' ', 'a', 'r', 'e', ' ', 'y', 'o',
'u', '\n']

In [2]:
```

Question8:

```
def load_items(filename):     try:
with open(filename, 'r') as file:
            items = {}
for line in file:
                name, price = line.strip().split('|')
items[name.strip()] = int(price.strip())
return items     except FileNotFoundError:
        print("VendingItems.txt not found.")
return {}

def vending_machine():
    items = load_items("VendingItems.txt")
if not items:
        return

    available_items = list(items.keys())

    # Step 1: Get a valid item from user
    while True:
        item = input("Enter item name: ").strip()
if item in items:
            break
else:
            print(f"Available Items are {available_items}.")
print("Try Again.")

    # Step 2: Get a valid amount from user
while True:
        money_input = input("Enter money: ").strip()
try:
            money = int(money_input)
break          except ValueError:
print(f"Bad Input {money_input}.")
print("Try Again.")

    price = items[item]
if money >= price:
        print("Thank you for your purchase. Enjoy")
change = money - price          if change > 0:
            print(f"Do not forget to collect your change, {change} Rs.")
else:
        print("Not enough money. Please try again with sufficient funds.")

vending_machine()
```

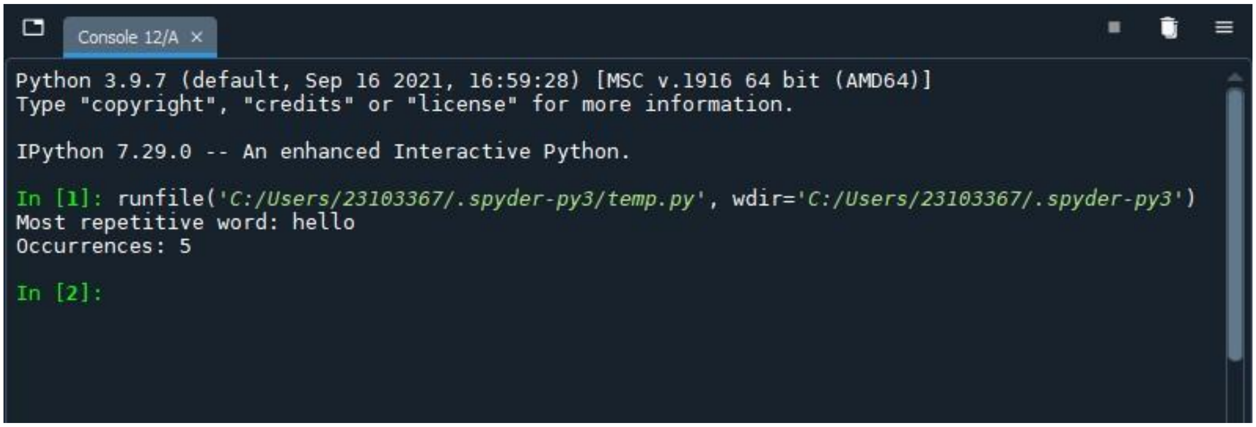


```
SyntaxError: invalid syntax


In [3]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')

Enter item name: Potato
Available Items are ['Potato Chips', 'Popcorn', 'Chocolate', 'Biscuit', 'Soft Drink'].
Try Again.

Enter item name: Potato Chips

Enter money: 20
Thank you for your purchase. Enjoy

In [4]:
```

Question10:

```
from collections import Counter

def most_repetitive_word(filename):
try:
        with open(filename, 'r') as file:
            text = file.read()             words = text.split()
word_counts = Counter(words)             most_common =
word_counts.most_common(1)             if
most_common:
                return most_common[0]  # (word, count)
else:
```

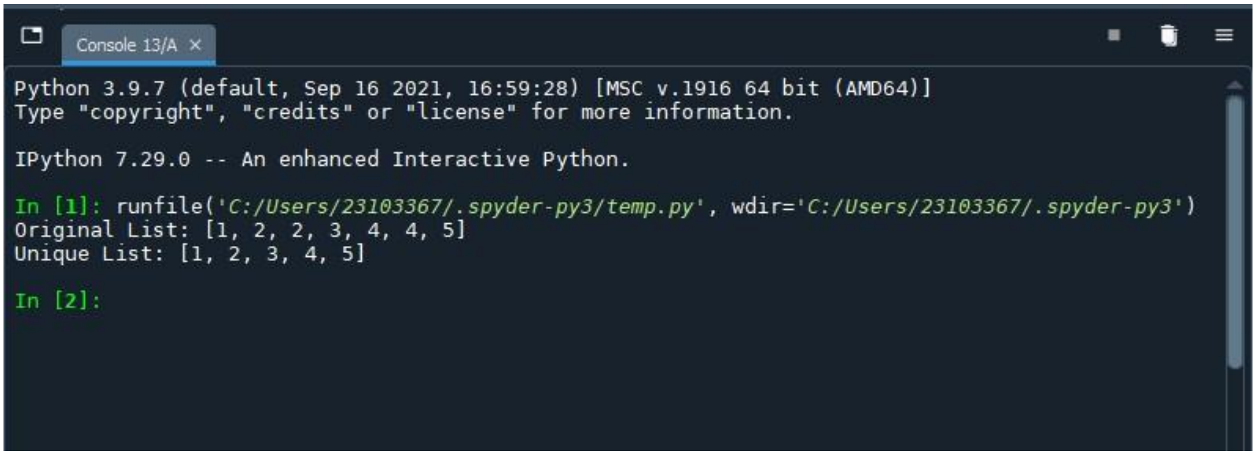
```
            return ("No words found", 0)
except FileNotFoundError:
return ("File not found", 0)

# Example usage:
result = most_repetitive_word("sample.txt")
print("Most repetitive word:", result[0])
print("Occurrences:", result[1])
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
Most repetitive word: hello
Occurrences: 5

In [2]:
```

Question11: def

```
unique_list(lst):
return list(set(lst))

# Example usage:
data = [1, 2, 2, 3, 4, 4, 5] result
= unique_list(data)
print("Original List:", data)
print("Unique List:", result)
```

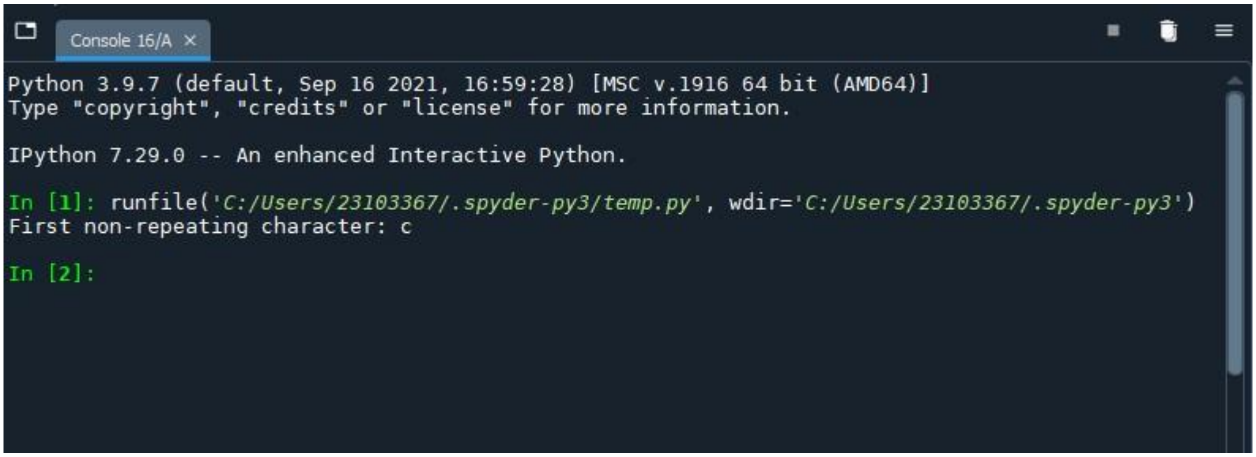


```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
Original List: [1, 2, 2, 3, 4, 4, 5]
Unique List: [1, 2, 3, 4, 5]

In [2]:
```

Question12:

```
from collections import OrderedDict
```

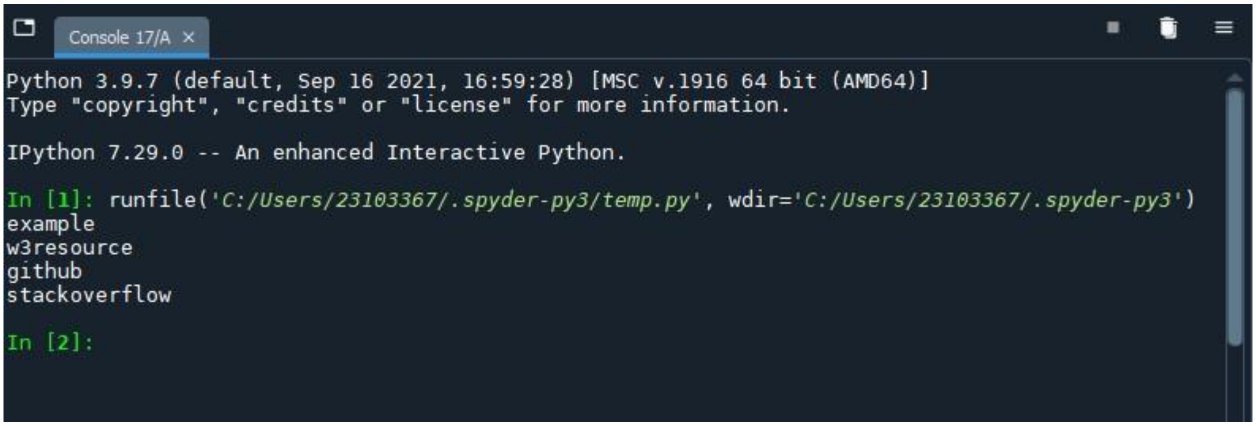
```
def first_non_repeating_char(s):
    count = OrderedDict()

    for char in s:
        count[char] = count.get(char, 0) + 1

    for char, freq in count.items():
        if freq == 1:
return char

    return None  # All characters are repeating

# Example usage:
input_str = "aabbcddee" result =
first_non_repeating_char(input_str) if result:
print("First non-repeating character:", result)
else:
    print("No non-repeating character found.")
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
First non-repeating character: c

In [2]:
```

Question13:

```
import re

def remove_parenthesis_area(data):
    return [re.sub(r'\s*\(.*?\)', '', item) for item in data]

# Sample input input_data = ["example (.com)", "w3resource", "github (.com)",
"stackoverflow (.com)"] output = remove_parenthesis_area(input_data)

# Print the result for
item in output:
print(item)
```



```
Python 3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.29.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/23103367/.spyder-py3/temp.py', wdir='C:/Users/23103367/.spyder-py3')
example
w3resource
github
stackoverflow

In [2]:
```