Classes:

1) Write a python class to convert an integer into a roman numeral and viceversa

2) Write a Python class to find validity of a string of parentheses, '(', ')', '{', '}', '[' and ']. These brackets must be close in the correct order, for example "()" and "()[]{}" are valid but "[)", "({[)]" and "{{{" are invalid.

3) Write a Python class to get all possible unique subsets from a set of distinct integers Input : [4, 5, 6] Output : [[], [6], [5], [5, 6], [4], [4, 6], [4, 5], [4, 5, 6]]
ANS:

```
class SubsetGenerator:
    def __init__(self):
        self.subsets = []

    def generate_subsets(self, nums):
        self._backtrack(nums, 0, [])
        return self.subsets

    def _backtrack(self, nums, start, path):
        self.subsets.append(path[:])

        for i in range(start, len(nums)):
            path.append(nums[i])
            self._backtrack(nums, i + 1, path)
            path.pop()

subset_generator = SubsetGenerator()
input_set = [4, 5, 6]
result = subset_generator.generate_subsets(input_set)
print("All possible unique subsets:")
print(result)
```

```python
class SubsetGenerator:
    def __init__(self):
        self.subsets = []

    def generate_subsets(self, nums):
        self._backtrack(nums, 0, [])
        return self.subsets

    def _backtrack(self, nums, start, path):
        self.subsets.append(path[:])

        for i in range(start, len(nums)):
            path.append(nums[i])
            self._backtrack(nums, i + 1, path)
            path.pop()

subset_generator = SubsetGenerator()
input_set = [4, 5, 6]
result = subset_generator.generate_subsets(input_set)
print("All possible unique subsets:")
print(result)
```

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Docum
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\
.py'
All possible unique subsets:
[[], [4], [4, 5], [4, 5, 6], [4, 6], [5], [5, 6], [6]]
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

4) Write a Python class to find a pair of elements (indices of the two numbers) from a given array whose sum equals a specific target number. Note: There will be one solution for each input and do not use the same element twice. Input: numbers= [90, 20,10,40,50,60,70], target=50 Output: 3, 4

ANS:

```python
class TwoSumFinder:
    def find_indices(self, numbers, target):
        num_index_mapping = {}
        for i, num in enumerate(numbers):
            complement = target - num
            if complement in num_index_mapping:
                return num_index_mapping[complement], i
            num_index_mapping[num] = i
        return None

two_sum_finder = TwoSumFinder()

numbers = [90, 20, 10, 40, 50, 60, 70]
target = 50

indices = two_sum_finder.find_indices(numbers, target)
```

```
if indices:
    print(f"The pair indices that sum to {target} are: {indices[0]}, {indices[1]}")
else:
    print("No such pair found.")
```

5) Write a Python class to find the three elements that sum to zero from a set of n real numbers. Input array : [-25, -10, -7, -3, 2, 4, 8, 10] Output : [[-10, 2, 8], [-7, -3, 10]]

6) Write a Python class to implement pow(x, n)
ANS:

```
class PowerCalculator:
    def pow(self, x, n):
        if n == 0:
            return 1
        if n < 0:
            return 1 / self.pow(x, -n)
        half_pow = self.pow(x, n // 2)
        if n % 2 == 0:
            return half_pow * half_pow
```

```
        else:
            return half_pow * half_pow * x

power_calculator = PowerCalculator()

x = float(input("Enter the value of x: "))
n = int(input("Enter the value of n: "))
result = power_calculator.pow(x, n)
print("{x} raised to the power of {n} is: {result}".format(x=x, n=n, result=result))
```

Welcome | Untitled-1.py

Untitled-1.py > ...

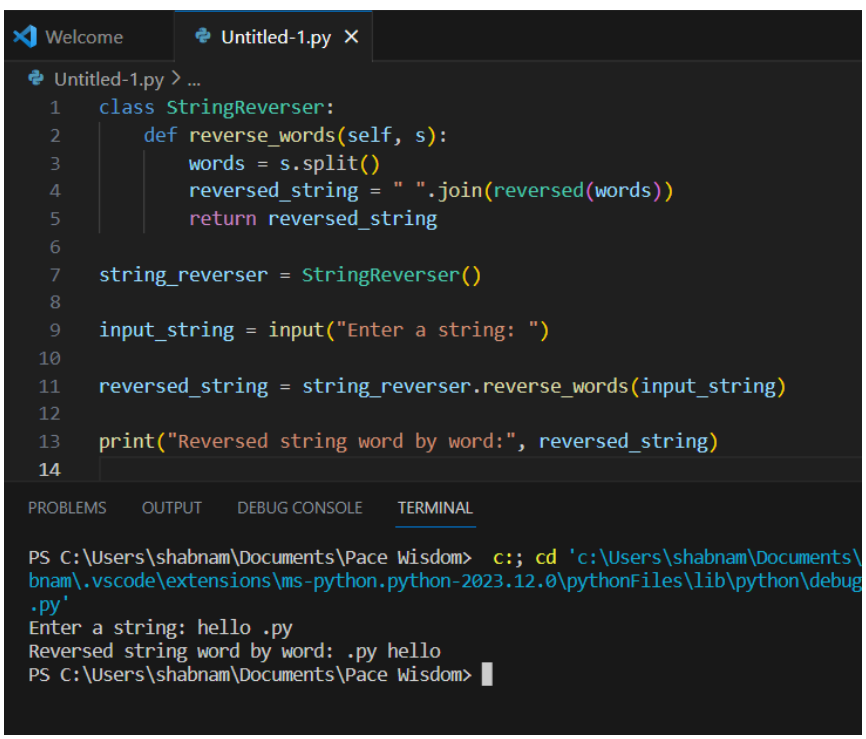
```
class PowerCalculator:
    def pow(self, x, n):
        if n == 0:
            return 1
        if n < 0:
            return 1 / self.pow(x, -n)
        half_pow = self.pow(x, n // 2)
        if n % 2 == 0:
            return half_pow * half_pow
        else:
            return half_pow * half_pow * x

power_calculator = PowerCalculator()

x = float(input("Enter the value of x: "))
n = int(input("Enter the value of n: "))
result = power_calculator.pow(x, n)
print("{x} raised to the power of {n} is: {result}".format(x=x, n=n, result=result))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace Wisdom'; & 'C:\Users
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\
.py'
Enter the value of x: 3
Enter the value of n: 2
3.0 raised to the power of 2 is: 9.0
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

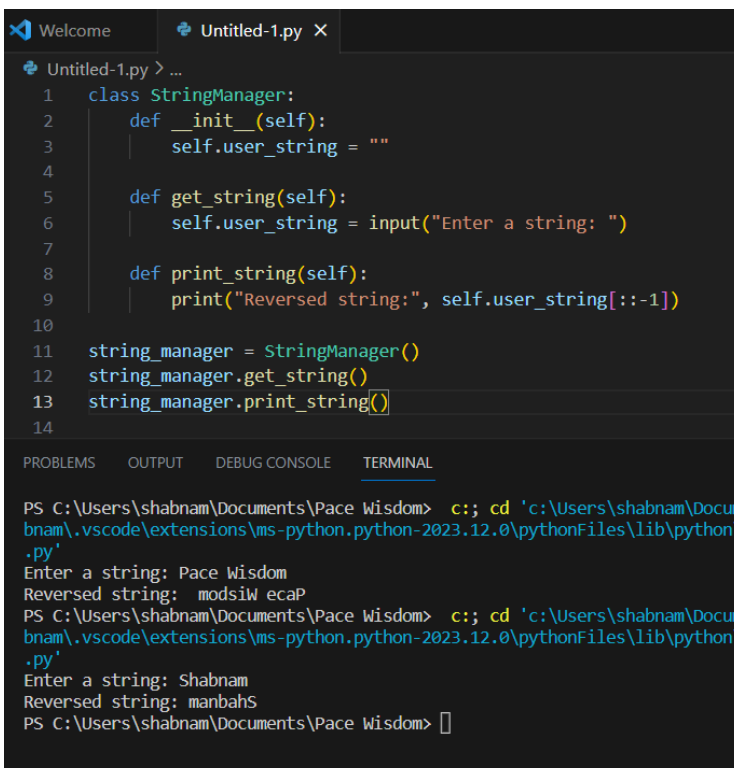
7) Write a Python class to reverse a string word by word.
Input string : 'hello .py' Expected Output : '.py hello'
ANS:

```
class StringReverser:
    def reverse_words(self, s):
        words = s.split()
        reversed_string = " ".join(reversed(words))
        return reversed_string

string_reverser = StringReverser()

input_string = input("Enter a string: ")

reversed_string = string_reverser.reverse_words(input_string)

print("Reversed string word by word:", reversed_string)
```



8) Write a python class which has 2 methods get_string and print_string. get_string takes a string from the user and print_string prints the string in reverse order.
ANS:

```
class StringManager:
    def __init__(self):
        self.user_string = ""

    def get_string(self):
        self.user_string = input("Enter a string: ")

    def print_string(self):
        print("Reversed string:", self.user_string[::-1])

string_manager = StringManager()
string_manager.get_string()
string_manager.print_string()
```



9) Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.
ANS:

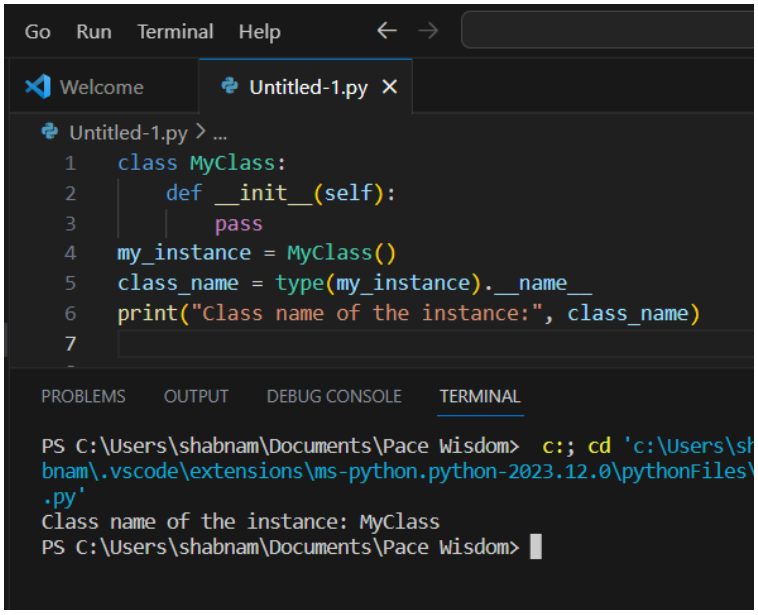
```
class Circle:
    def __init__(self, radius):
        self.radius = radius
```

```
    def area(self):
        return 3.14159 * self.radius**2

    def perimeter(self):
        return 2 * 3.14159 * self.radius
radius = float(input("Enter the radius of the circle: "))
circle = Circle(radius)
print("Area of the circle:", circle.area())
print("Perimeter of the circle:", circle.perimeter())
```

10) Write a Python program to get the class name of an instance in Python.

ANS:

```
class MyClass:
    def __init__(self):
        pass
my_instance = MyClass()
class_name = type(my_instance).__name__
print("Class name of the instance:", class_name)
```



Lambda:

1) Write a Python program to create a lambda function that adds 15 to a given number passed in as an argument, also create a lambda function that multiplies argument x with argument y and prints the result.
 Sample Output: 25 48

ANS:

```
add_15 = lambda x: x + 15
multiply = lambda x, y: x * y

number = int(input("Enter a number: "))
x = int(input("Enter the first number for multiplication: "))
y = int(input("Enter the second number for multiplication: "))

result_add = add_15(number)
result_multiply = multiply(x, y)

print("Result of adding 15 to the number:", result_add)
print("Result of multiplying", x, "and", y, "is:", result_multiply)
```

Welcome Untitled-1.py

Untitled-1.py > ...

```python
add_15 = lambda x: x + 15
multiply = lambda x, y: x * y

number = int(input("Enter a number: "))
x = int(input("Enter the first number for multiplication: "))
y = int(input("Enter the second number for multiplication: "))

result_add = add_15(number)
result_multiply = multiply(x, y)

print("Result of adding 15 to the number:", result_add)
print("Result of multiplying", x, "and", y, "is:", result_multiply)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\a
.py'
Enter a number: 27
Enter the first number for multiplication: 5
Enter the second number for multiplication: 6
Result of adding 15 to the number: 42
Result of multiplying 5 and 6 is: 30
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

2) Write a Python program to sort a list of tuples using Lambda.
Original list of tuples: [('English', 88), ('Science', 90), ('Maths', 97), ('Social sciences', 82)]
Sorting the List of Tuples: [('Social sciences', 82), ('English', 88), ('Science', 90), ('Maths', 97)]
ANS:

```python
n = int(input("Enter the number of tuples: "))
original_list = []
for i in range(n):
    subject = input("Enter the subject name: ")
    score = int(input("Enter the score: "))
    original_list.append((subject, score))

sorted_list = sorted(original_list, key=lambda x: x[1])

print("Sorting the List of Tuples:", sorted_list)
```

```python
n = int(input("Enter the number of tuples: "))
original_list = []
for i in range(n):
    subject = input("Enter the subject name: ")
    score = int(input("Enter the score: "))
    original_list.append((subject, score))

sorted_list = sorted(original_list, key=lambda x: x[1])

print("Sorting the List of Tuples:", sorted_list)
```

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace Wisdom'; & 'C:\Us
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debug
.py'
Enter the number of tuples: 4
Enter the subject name: Math
Enter the score: 23
Enter the subject name: Science
Enter the score: 6
Enter the subject name: Social Science
Enter the score: 19
Enter the subject name: Hindi
Enter the score: 20
Sorting the List of Tuples: [('Science', 6), ('Social Science', 19), ('Hindi', 20), ('Math', 23)]
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

3) Write a Python program to sort a list of dictionaries using Lambda.
Original list of dictionaries : [{'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Mi Max', 'model': '2', 'color': 'Gold'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}]
Sorting the List of dictionaries : [{'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}, {'make': 'Mi Max', 'model': '2', 'color': 'Gold'}]
ANS:

```python
original_list = [{'make': 'Nokia', 'model': 216, 'color': 'Black'},
                 {'make': 'Mi Max', 'model': '2', 'color': 'Gold'},
                 {'make': 'Samsung', 'model': 7, 'color': 'Blue'}]

sorted_list = sorted(original_list, key=lambda x: x['make'])
print("Sorting the List of dictionaries:", sorted_list)
```

```python
original_list = [{'make': 'Nokia', 'model': 216, 'color': 'Black'},
                 {'make': 'Mi Max', 'model': '2', 'color': 'Gold'},
                 {'make': 'Samsung', 'model': 7, 'color': 'Blue'}]

sorted_list = sorted(original_list, key=lambda x: x['make'])
print("Sorting the List of dictionaries:", sorted_list)
```

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace Wisdom'; & 'C:\Users\shabnam\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\shabnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '58479' '--' 'c:\Users\shabnam\Documents\Pace Wisdom\Untitled-1.py'
Sorting the List of dictionaries: [{'make': 'Mi Max', 'model': '2', 'color': 'Gold'}, {'make': 'Nokia', 'model': 216, 'color': 'Black'}, {'make': 'Samsung', 'model': 7, 'color': 'Blue'}]
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

5) Write a Python program to check whether a given string is a number or not using Lambda.
ANS:

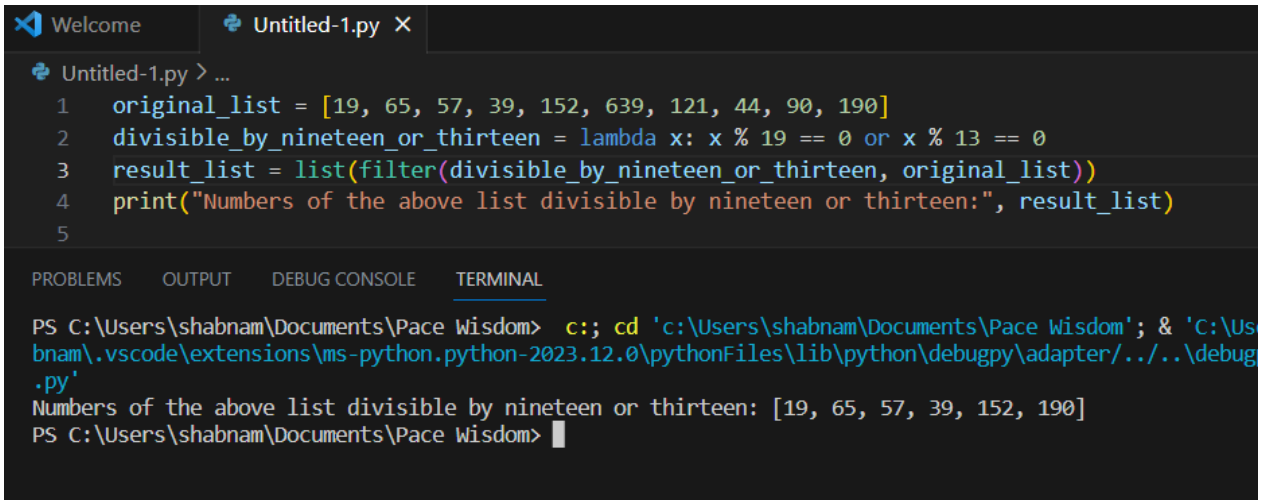
```python
is_number = lambda s: s.isdigit()
input_string = input("Enter a string: ")

if is_number(input_string):
    print("The string '{}' is a number.".format(input_string))
else:
    print("The string '{}' is not a number.".format(input_string))
```

```python
is_number = lambda s: s.isdigit()
input_string = input("Enter a string: ")

if is_number(input_string):
    print("The string '{}' is a number.".format(input_string))
else:
    print("The string '{}' is not a number.".format(input_string))
```

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pac
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\
.py'
Enter a string: 567
The string '567' is a number.
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pac
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\
.py'
Enter a string: Shabnam
The string 'Shabnam' is not a number.
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

6) Write a Python program to find numbers divisible by nineteen or thirteen from a list of numbers using Lambda
Original list: [19, 65, 57, 39, 152, 639, 121, 44, 90, 190]
Numbers of the above list divisible by nineteen or thirteen: [19, 65, 57, 39, 152, 190]
ANS:

```
original_list = [19, 65, 57, 39, 152, 639, 121, 44, 90, 190]
divisible_by_nineteen_or_thirteen = lambda x: x % 19 == 0 or x % 13 == 0
result_list = list(filter(divisible_by_nineteen_or_thirteen, original_list))
print("Numbers of the above list divisible by nineteen or thirteen:", result_list)
```

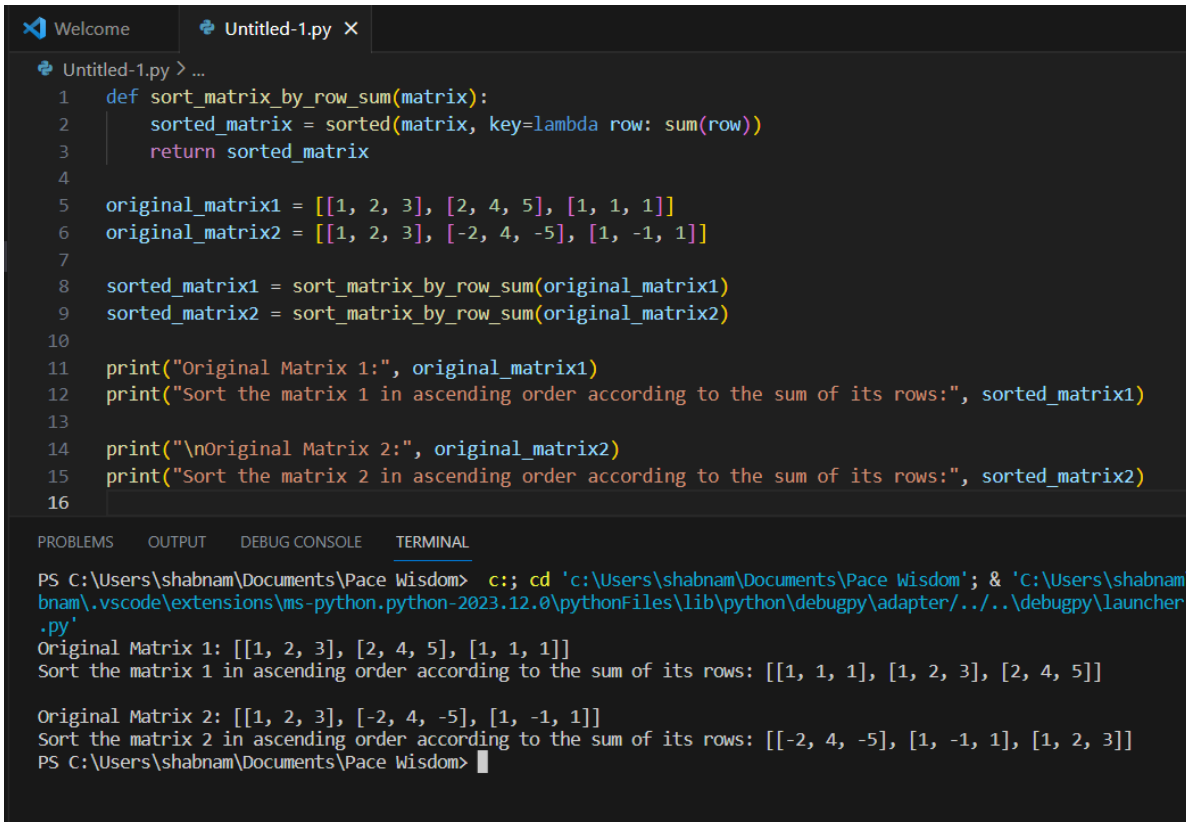


7) Write a Python program to sort a given matrix in ascending order according to the sum of its rows using lambda.
Original Matrix: [[1, 2, 3], [2, 4, 5], [1, 1, 1]]
Sort the said matrix in ascending order according to the sum of its rows [[1, 1, 1], [1, 2, 3], [2, 4, 5]]
Original Matrix: [[1, 2, 3], [-2, 4, -5], [1, -1, 1]]
Sort the said matrix in ascending order according to the sum of its rows [[-2, 4, -5], [1, -1, 1], [1, 2, 3]]
ANS:

```
def sort_matrix_by_row_sum(matrix):
    sorted_matrix = sorted(matrix, key=lambda row: sum(row))
    return sorted_matrix

original_matrix1 = [[1, 2, 3], [2, 4, 5], [1, 1, 1]]
original_matrix2 = [[1, 2, 3], [-2, 4, -5], [1, -1, 1]]

sorted_matrix1 = sort_matrix_by_row_sum(original_matrix1)
sorted_matrix2 = sort_matrix_by_row_sum(original_matrix2)

print("Original Matrix 1:", original_matrix1)
```

print("Sort the matrix 1 in ascending order according to the sum of its rows:", sorted_matrix1)

print("\nOriginal Matrix 2:", original_matrix2)
print("Sort the matrix 2 in ascending order according to the sum of its rows:", sorted_matrix2)



8) Write a Python program to check whether a given string contains a capital letter, a lower case letter, a number and a minimum length using lambda. Minimum length : 10 input string: PaceWisd0m o/p: valid string

9) Write a Python program to find the elements of a given list of strings that contain specific substring using lambda.
Original list: ['red', 'black', 'white', 'green', 'orange']
Substring to search: ack Elements of the said list that contain specific substring: ['black']
Substring to search: abc Elements of the said list that contain specific substring: []
ANS:
original_list = ['red', 'black', 'white', 'green', 'orange']
contains_substring = lambda s, substring: substring in s
substring_to_search = input("Substring to search: ")

filtered_list = list(filter(lambda s: contains_substring(s, substring_to_search), original_list))
print("Elements of the list that contain the specific substring:", filtered_list)

```python
original_list = ['red', 'black', 'white', 'green', 'orange']
contains_substring = lambda s, substring: substring in s
substring_to_search = input("Substring to search: ")

filtered_list = list(filter(lambda s: contains_substring(s, substring_to_search), original_list))
print("Elements of the list that contain the specific substring:", filtered_list)
```

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace Wisdom'; & 'C:\Users\shabnam\AppD
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '58
.py'
Substring to search: ack
Elements of the list that contain the specific substring: ['black']
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace Wisdom'; & 'C:\Users\shabnam\AppD
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '58
.py'
Substring to search: te
Elements of the list that contain the specific substring: ['white']
PS C:\Users\shabnam\Documents\Pace Wisdom>
```

10) Write a Python program to sort a given mixed list of integers and strings using lambda. Numbers must be sorted before strings.
Original list: [19, 'red', 12, 'green', 'blue', 10, 'white', 'green', 1]
Sort the said mixed list of integers and strings: [1, 10, 12, 19, 'blue', 'green', 'green', 'red', 'white']
ANS:
original_list = [19, 'red', 12, 'green', 'blue', 10, 'white', 'green', 1]
sort_order = lambda x: (isinstance(x, int), x)
sorted_list = sorted(original_list, key=sort_order)
print("Sort the mixed list of integers and strings:", sorted_list)

```python
original_list = [19, 'red', 12, 'green', 'blue', 10, 'white', 'green', 1]
sort_order = lambda x: (isinstance(x, int), x)
sorted_list = sorted(original_list, key=sort_order)
print("Sort the mixed list of integers and strings:", sorted_list)
```

```
PS C:\Users\shabnam\Documents\Pace Wisdom>  c:; cd 'c:\Users\shabnam\Documents\Pace Wisdom'; & 'C:\Users\s
bnam\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\la
.py'
Sort the mixed list of integers and strings: ['blue', 'green', 'green', 'red', 'white', 1, 10, 12, 19]
PS C:\Users\shabnam\Documents\Pace Wisdom>
```