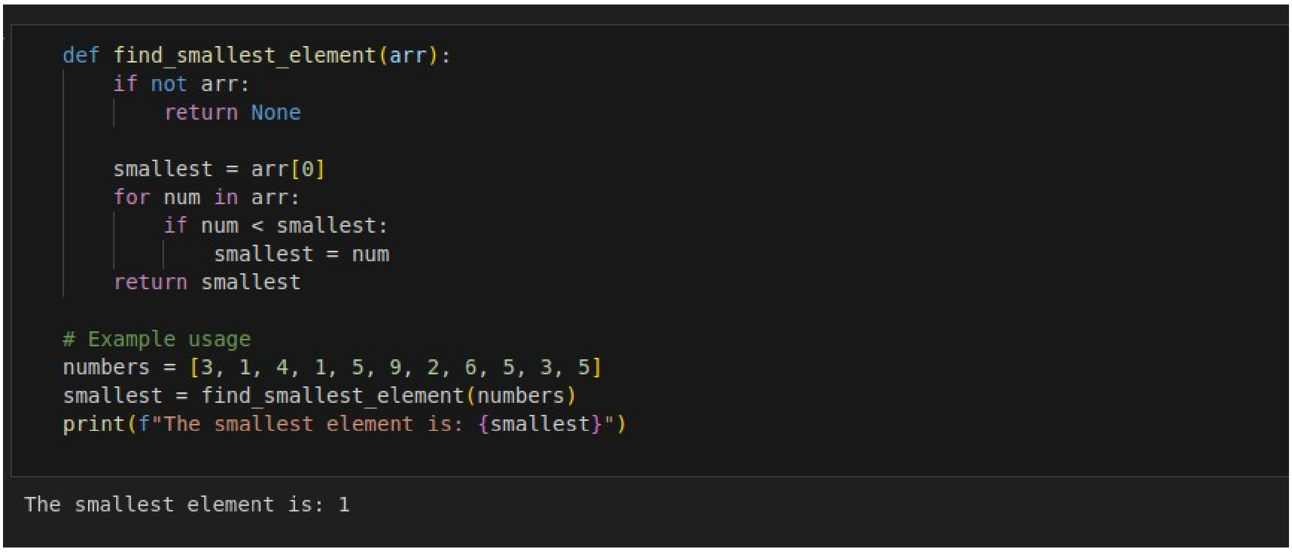
Lab-4 Python Programming Basics Assignment

Name: Ketan Goud
Reg No: 220905260
Roll No: 39
Section: D D2

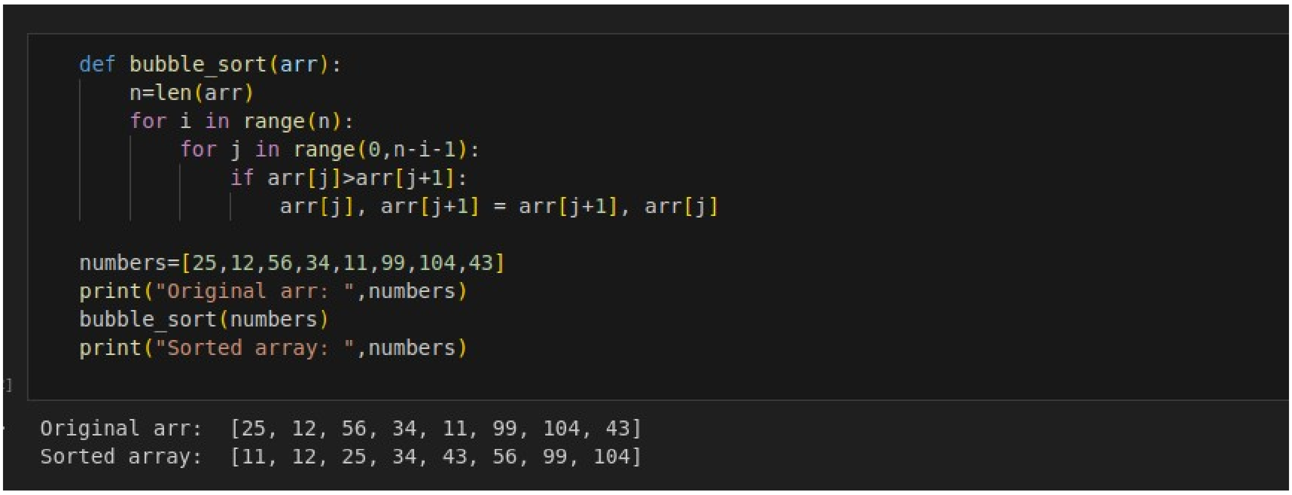
1. Write a python program to select smallest element from a list in an expected linear time.



```
def find_smallest_element(arr):
    if not arr:
        return None

    smallest = arr[0]
    for num in arr:
        if num < smallest:
            smallest = num
    return smallest

# Example usage
numbers = [3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5]
smallest = find_smallest_element(numbers)
print(f"The smallest element is: {smallest}")
```

```
The smallest element is: 1
```

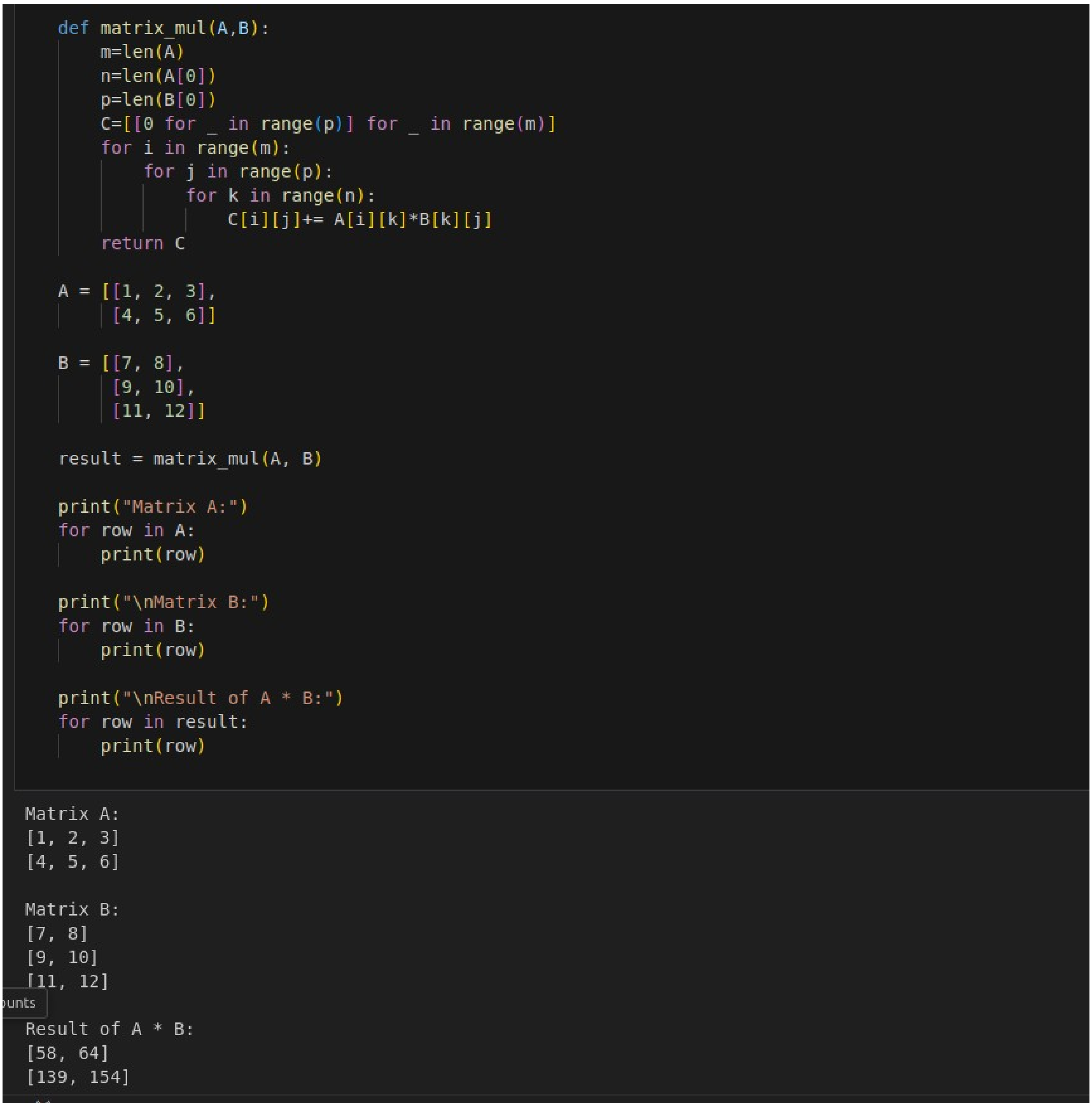
2. Write a python program to implement bubble sort.



```
def bubble_sort(arr):
    n=len(arr)
    for i in range(n):
        for j in range(0,n-i-1):
            if arr[j]>arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]

numbers=[25,12,56,34,11,99,104,43]
print("Original arr: ",numbers)
bubble_sort(numbers)
print("Sorted array: ",numbers)
```

```
Original arr:  [25, 12, 56, 34, 11, 99, 104, 43]
Sorted array:  [11, 12, 25, 34, 43, 56, 99, 104]
```

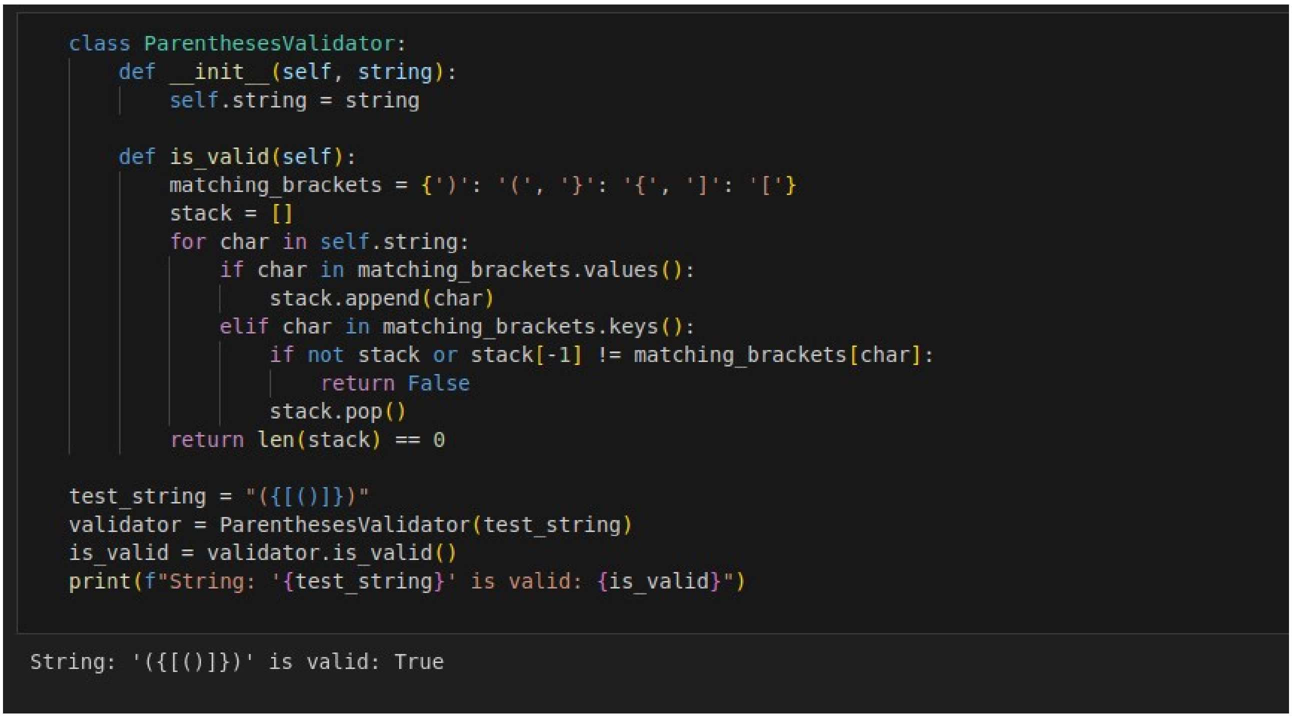
3. Write a python program to multiply two matrices



```
def matrix_mul(A,B):
    m=len(A)
    n=len(A[0])
    p=len(B[0])
    C=[[0 for _ in range(p)] for _ in range(m)]
    for i in range(m):
        for j in range(p):
            for k in range(n):
                C[i][j]+= A[i][k]*B[k][j]
    return C

A = [[1, 2, 3],
     [4, 5, 6]]

B = [[7, 8],
     [9, 10],
     [11, 12]]

result = matrix_mul(A, B)

print("Matrix A:")
for row in A:
    print(row)

print("\nMatrix B:")
for row in B:
    print(row)

print("\nResult of A * B:")
for row in result:
    print(row)
```

```
Matrix A:
[1, 2, 3]
[4, 5, 6]

Matrix B:
[7, 8]
[9, 10]
[11, 12]

Result of A * B:
[58, 64]
[139, 154]
```

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

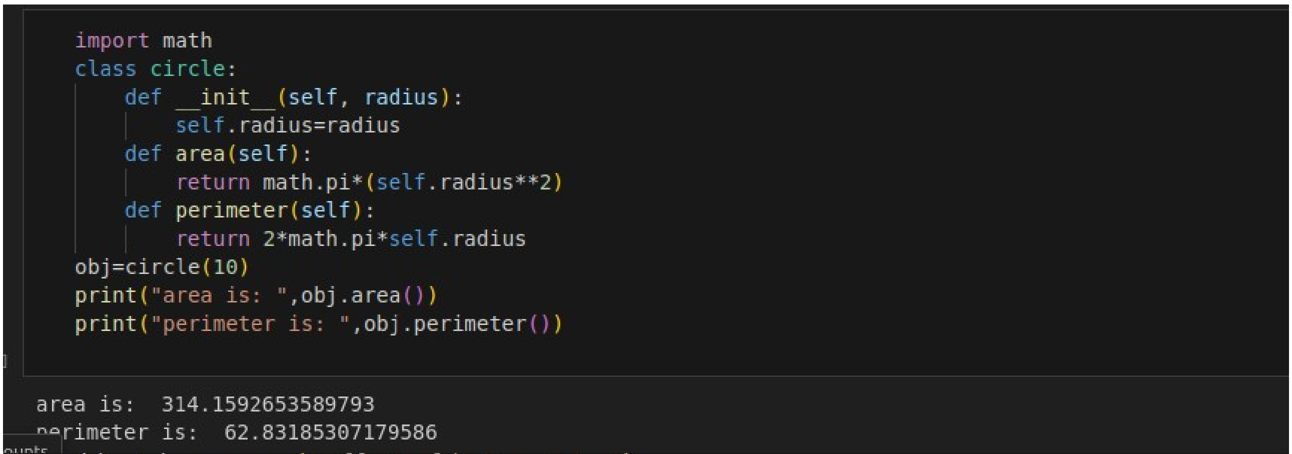


```
class ParenthesesValidator:
    def __init__(self, string):
        self.string = string

    def is_valid(self):
        matching_brackets = {')': '(', '}': '{', ']': '['}
        stack = []
        for char in self.string:
            if char in matching_brackets.values():
                stack.append(char)
            elif char in matching_brackets.keys():
                if not stack or stack[-1] != matching_brackets[char]:
                    return False
                stack.pop()
        return len(stack) == 0

test_string = "({[()]})"
validator = ParenthesesValidator(test_string)
is_valid = validator.is_valid()
print(f"String: '{test_string}' is valid: {is_valid}")
```

```
String: '({[()]})' is valid: True
```

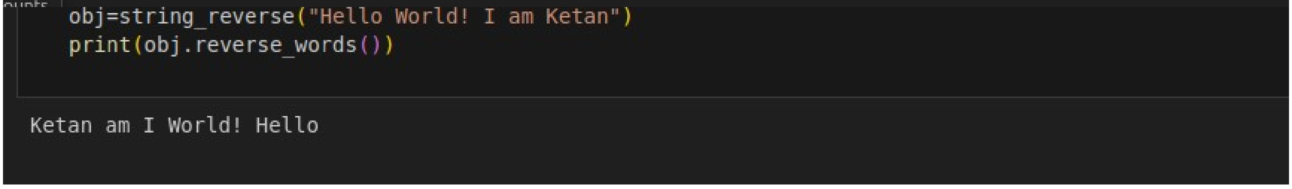
5. Write a Python class to reverse a string word by word.

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



```
import math
class circle:
    def __init__(self, radius):
        self.radius=radius
    def area(self):
        return math.pi*(self.radius**2)
    def perimeter(self):
        return 2*math.pi*self.radius
obj=circle(10)
print("area is: ",obj.area())
print("perimeter is: ",obj.perimeter())
```

```
area is:  314.1592653589793
perimeter is:  62.83185307179586
```



```
obj=string_reverse("Hello World! I am Ketan")
print(obj.reverse_words())
```

```
Ketan am I World! Hello
```