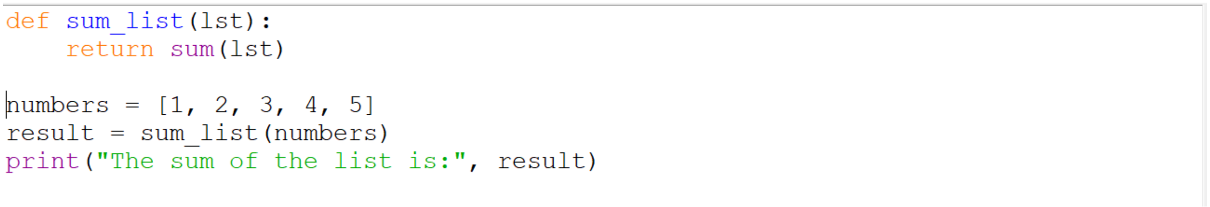
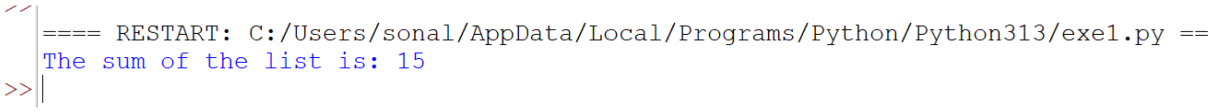
1. Write a function sum_list(lst) that takes a list of numbers and returns their sum.

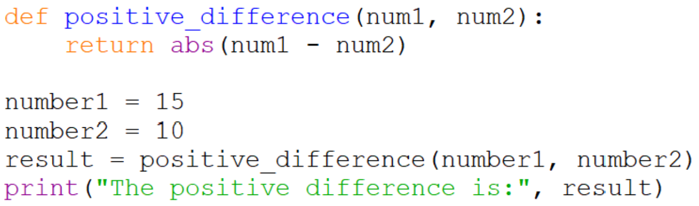


```
def sum_list(lst):
    return sum(lst)

numbers = [1, 2, 3, 4, 5]
result = sum_list(numbers)
print("The sum of the list is:", result)
```

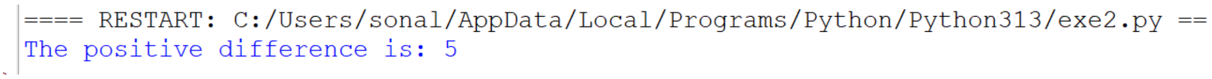


```
==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe1.py ==
The sum of the list is: 15
>>
```

2. Write a program to print the positive difference of two numbers.

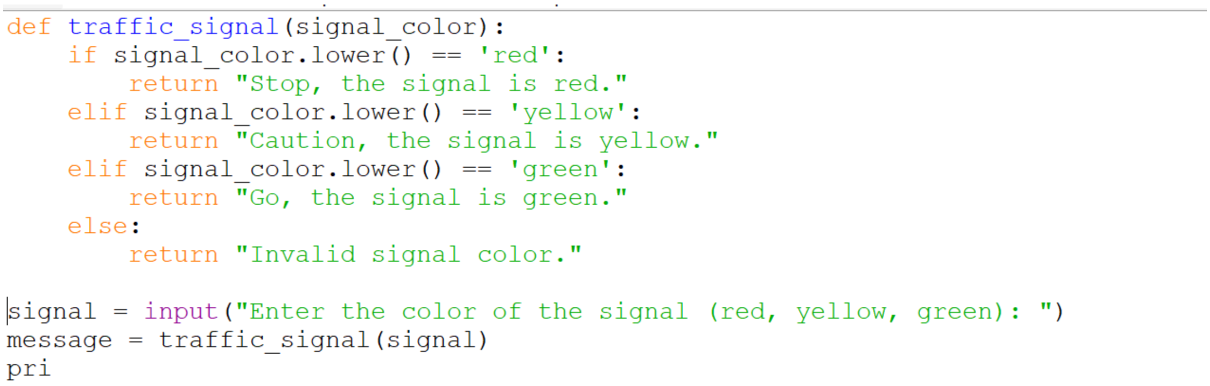


```
def positive_difference(num1, num2):
    return abs(num1 - num2)

number1 = 15
number2 = 10
result = positive_difference(number1, number2)
print("The positive difference is:", result)
```



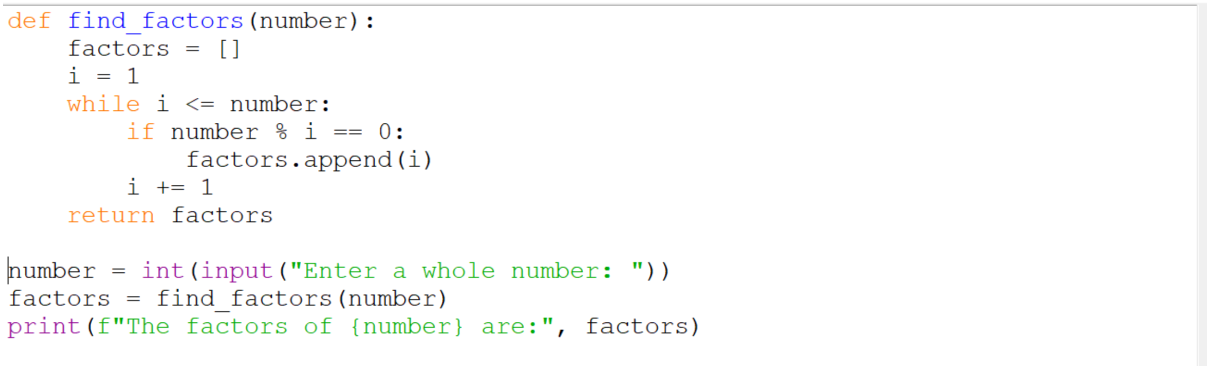
```
==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe2.py ==
The positive difference is: 5
```

3. Display the appropriate message as per the colour of signal at the road crossing.

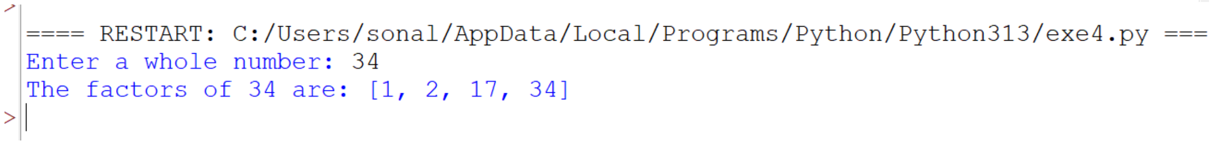


```
def traffic_signal(signal_color):
    if signal_color.lower() == 'red':
        return "Stop, the signal is red."
    elif signal_color.lower() == 'yellow':
        return "Caution, the signal is yellow."
    elif signal_color.lower() == 'green':
        return "Go, the signal is green."
    else:
        return "Invalid signal color."

signal = input("Enter the color of the signal (red, yellow, green): ")
message = traffic_signal(signal)
pri
```

4. Program to find the factors of a whole number using while loop.

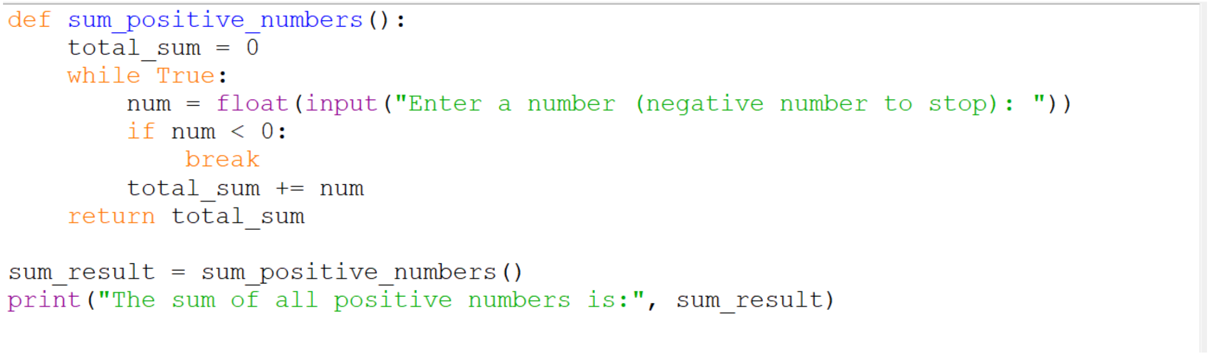


```
def find_factors(number):
    factors = []
    i = 1
    while i <= number:
        if number % i == 0:
            factors.append(i)
        i += 1
    return factors

number = int(input("Enter a whole number: "))
factors = find_factors(number)
print(f"The factors of {number} are:", factors)
```

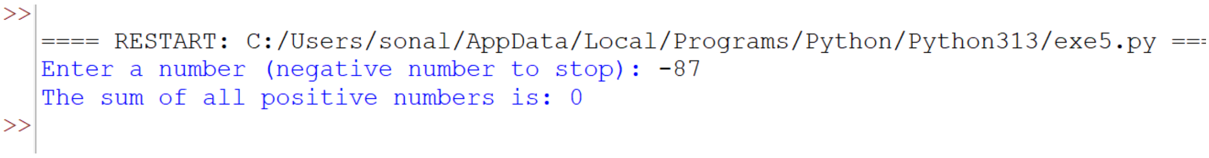


```
==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe4.py ===
Enter a whole number: 34
The factors of 34 are: [1, 2, 17, 34]
>
```

5. Find the sum of all the positive numbers entered by the user. As soon as the user enters a negative
number, stop taking in any further input from the user and display the sum.

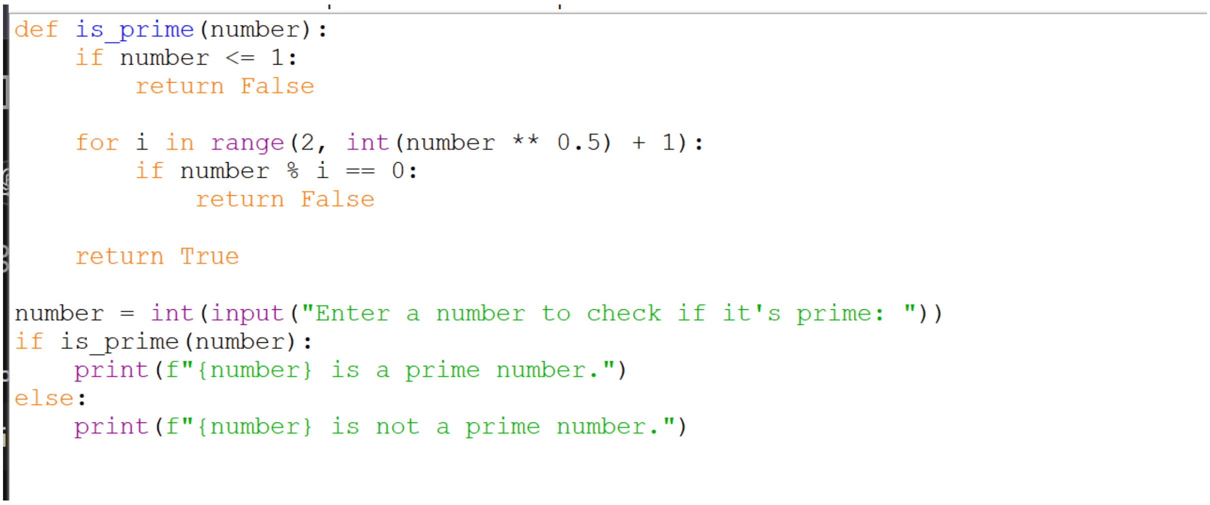


```
def sum_positive_numbers():
    total_sum = 0
    while True:
        num = float(input("Enter a number (negative number to stop): "))
        if num < 0:
            break
        total_sum += num
    return total_sum

sum_result = sum_positive_numbers()
print("The sum of all positive numbers is:", sum_result)
```



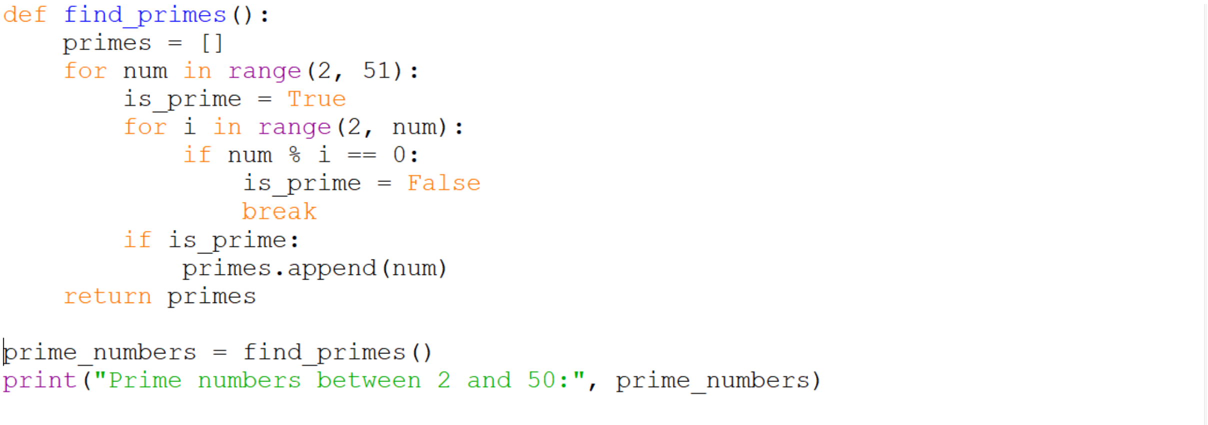
```
>>
   ==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe5.py ==
   Enter a number (negative number to stop): -87
   The sum of all positive numbers is: 0
>>
```

6.Program to check if the input number is prime or not.

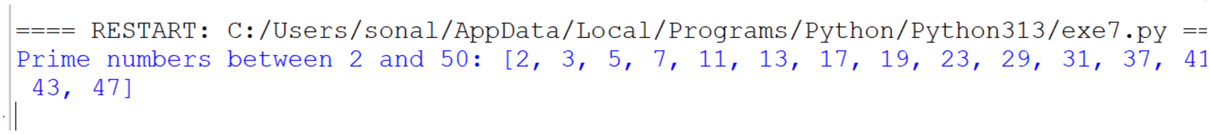


```
def is_prime(number):
    if number <= 1:
        return False

    for i in range(2, int(number ** 0.5) + 1):
        if number % i == 0:
            return False

    return True

number = int(input("Enter a number to check if it's prime: "))
if is_prime(number):
    print(f"{number} is a prime number.")
else:
    print(f"{number} is not a prime number.")
```

7. Program to find prime numbers between 2 to 50 using nested for loops.

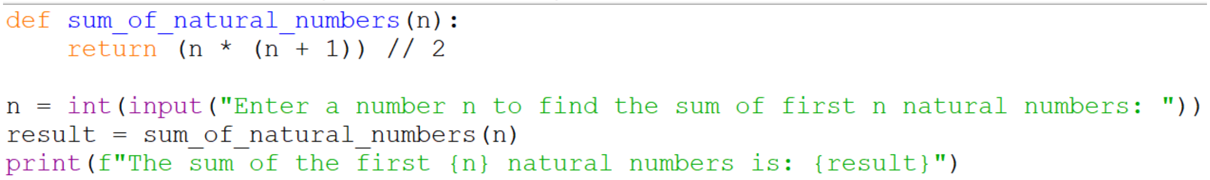


```
def find_primes():
    primes = []
    for num in range(2, 51):
        is_prime = True
        for i in range(2, num):
            if num % i == 0:
                is_prime = False
                break
        if is_prime:
            primes.append(num)
    return primes

prime_numbers = find_primes()
print("Prime numbers between 2 and 50:", prime_numbers)
```

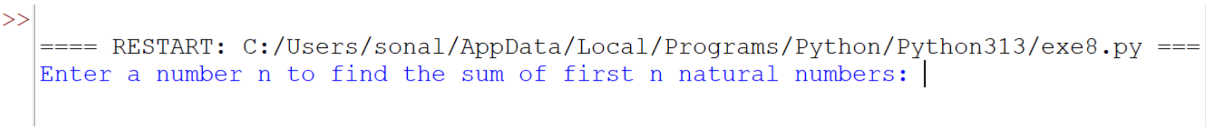


```
==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe7.py ==
Prime numbers between 2 and 50: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41
 43, 47]
```

8. Write a program using a user defined function that displays sum of first n natural numbers, where n
is passed as an argument.

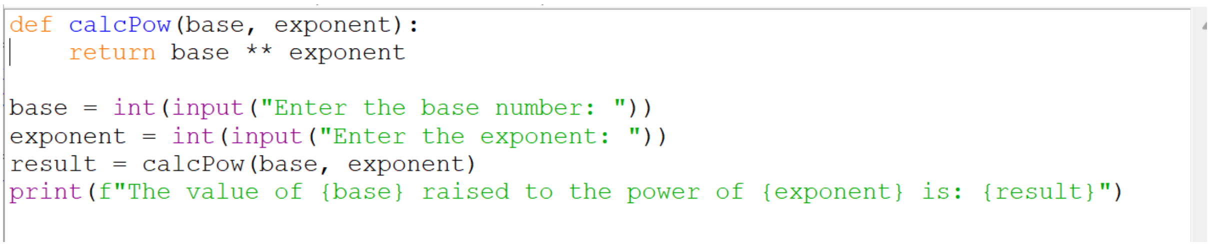


```
def sum_of_natural_numbers(n):
    return (n * (n + 1)) // 2

n = int(input("Enter a number n to find the sum of first n natural numbers: "))
result = sum_of_natural_numbers(n)
print(f"The sum of the first {n} natural numbers is: {result}")
```

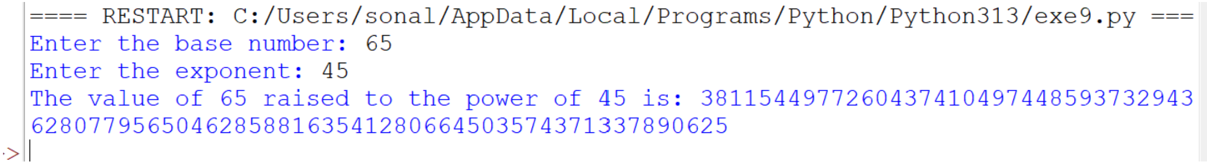


```
>>
 ==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe8.py ===
 Enter a number n to find the sum of first n natural numbers: 
```

9. Write a program using user defined function calcPow() that accepts base and exponent as arguments and returns the value Baseexponent where Base and exponent are integers.

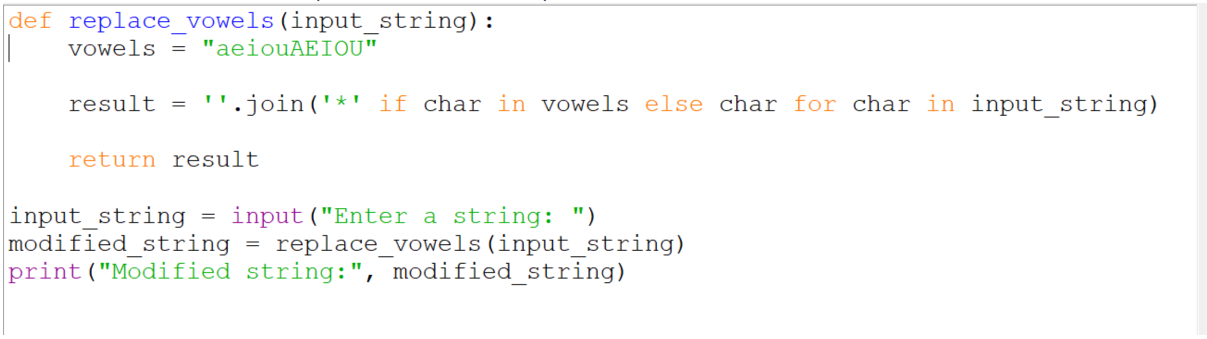


```
def calcPow(base, exponent):
    return base ** exponent

base = int(input("Enter the base number: "))
exponent = int(input("Enter the exponent: "))
result = calcPow(base, exponent)
print(f"The value of {base} raised to the power of {exponent} is: {result}")
```

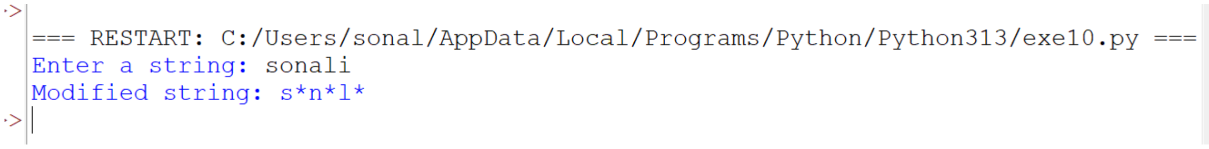


```
==== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe9.py ===
Enter the base number: 65
Enter the exponent: 45
The value of 65 raised to the power of 45 is: 38115449772604374104974485937329436
2807795650462858816354128066450357437133789 0625
>
```

10. Write a program with a user defined function with string as a parameter which replaces all vowels in
the string with &#39;*&#39;.

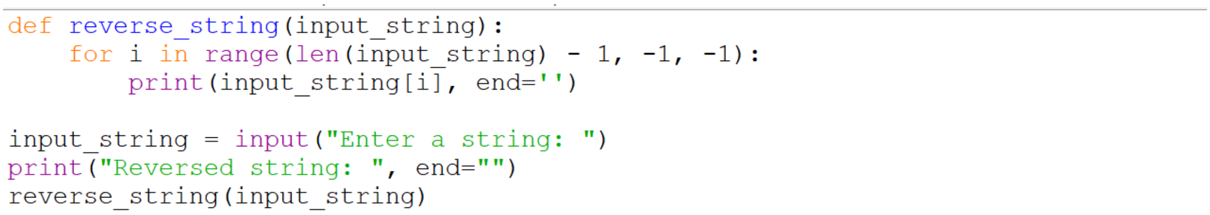


```
def replace_vowels(input_string):
    vowels = "aeiouAEIOU"

    result = ''.join('*' if char in vowels else char for char in input_string)

    return result

input_string = input("Enter a string: ")
modified_string = replace_vowels(input_string)
print("Modified string:", modified_string)
```

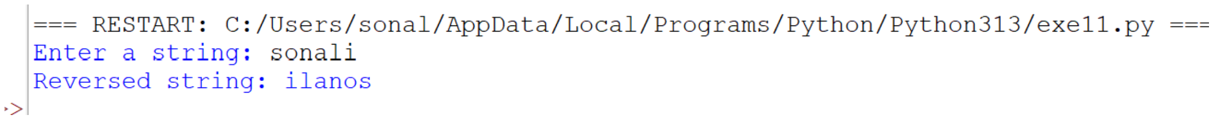


```
>
 === RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe10.py ===
 Enter a string: sonali
 Modified string: s*n*l*
>
```

11. Write a program to input a string from the user and print it in the reverse order without creating a
new string.

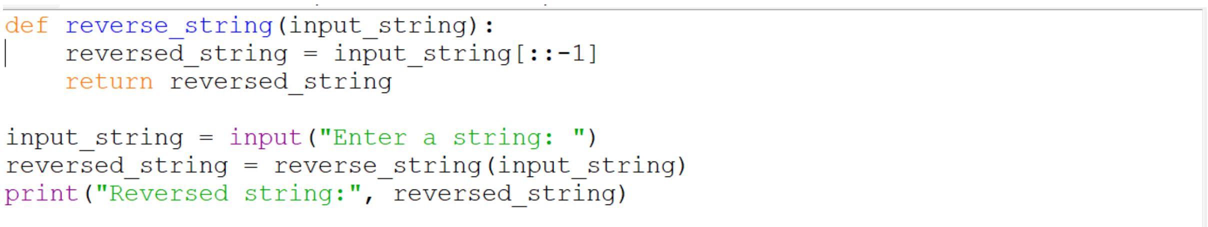


```
def reverse_string(input_string):
    for i in range(len(input_string) - 1, -1, -1):
        print(input_string[i], end='')

input_string = input("Enter a string: ")
print("Reversed string: ", end="")
reverse_string(input_string)
```

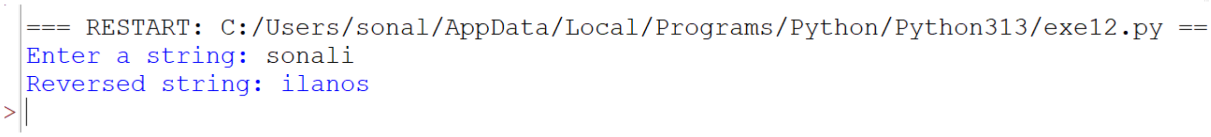


```
=== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe11.py ===
Enter a string: sonali
Reversed string: ilanos
>
```

12. Write a program which reverses a string passed as parameter and stores the reversed string in a
new string. Use a user defined function for reversing the string.



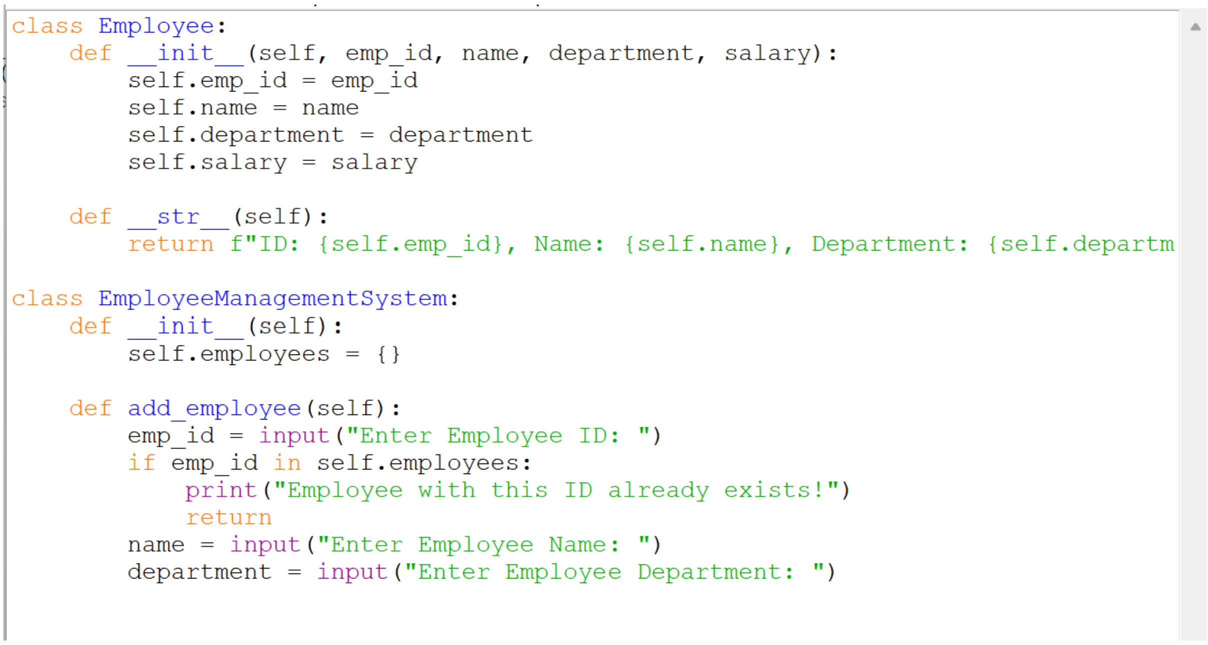
```
def reverse_string(input_string):
    reversed_string = input_string[::-1]
    return reversed_string

input_string = input("Enter a string: ")
reversed_string = reverse_string(input_string)
print("Reversed string:", reversed_string)
```



```
=== RESTART: C:/Users/sonal/AppData/Local/Programs/Python/Python313/exe12.py ==
Enter a string: sonali
Reversed string: ilanos
>
```

19. Create a menu-driven Employee Management System using Python classes and objects. The system
will allow the user to:

1. Add new employees with details.
2. View all employees.
3. Update employee information.
4. Delete an employee record.
5. Exit the program.



```
class Employee:
    def __init__(self, emp_id, name, department, salary):
        self.emp_id = emp_id
        self.name = name
        self.department = department
        self.salary = salary

    def __str__(self):
        return f"ID: {self.emp_id}, Name: {self.name}, Department: {self.departm

class EmployeeManagementSystem:
    def __init__(self):
        self.employees = {}

    def add_employee(self):
        emp_id = input("Enter Employee ID: ")
        if emp_id in self.employees:
            print("Employee with this ID already exists!")
            return
        name = input("Enter Employee Name: ")
        department = input("Enter Employee Department: ")
```