# INSTITUTE OF COMPUTER TECHNOLOGY
# B-TECH COMPUTER SCIENCE ENGINEERING 2025-26
# SUBJECT: ALGORITHM ANALYSIS AND DESIGN

NAME: Rahul Prajapati

ENRLL. NO: 23162171020

BRANCH: CYBER SECURITY

BATCH: 52

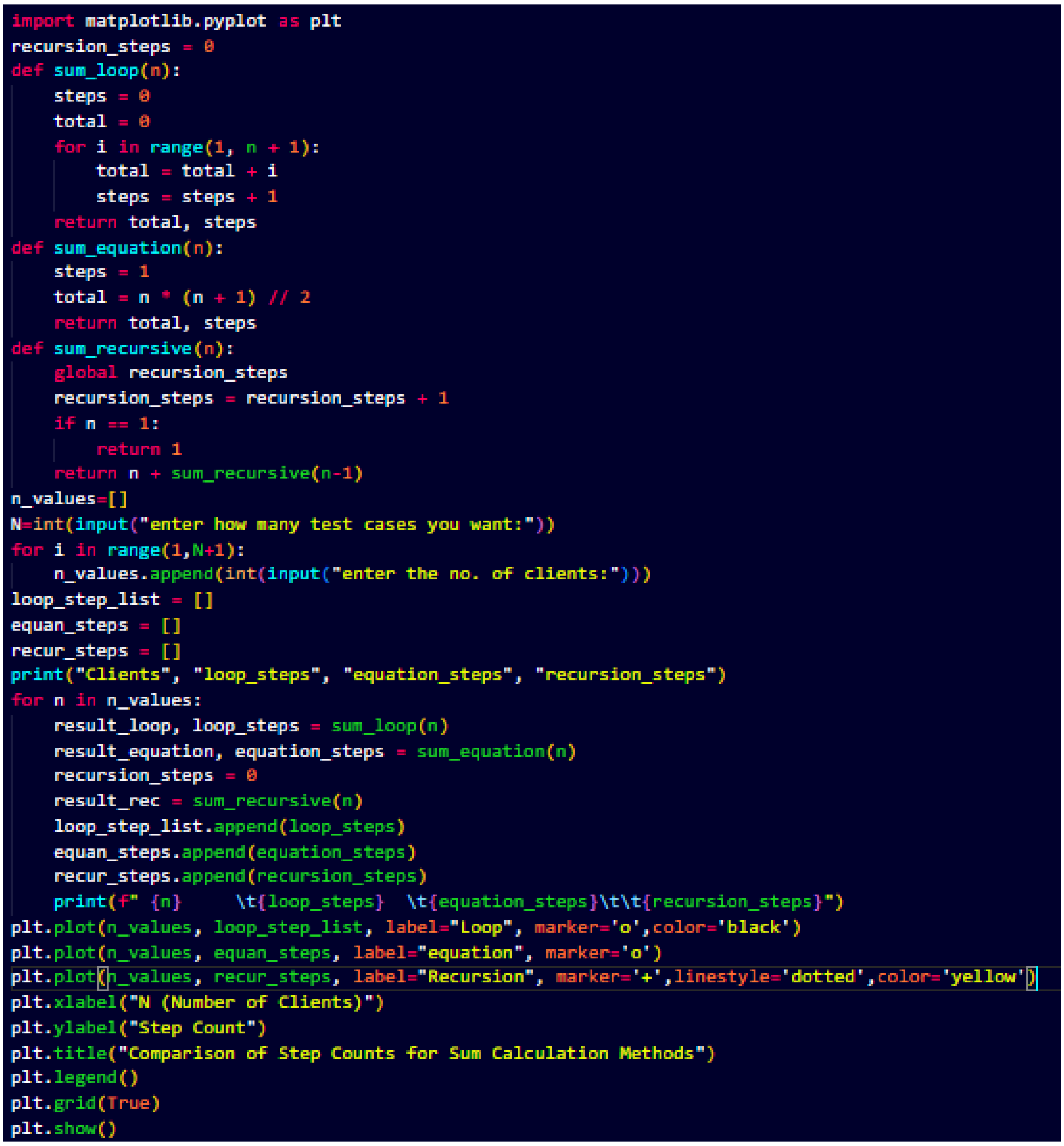
## PRACTICAL_02

1. MPSoft Technologies Pvt. Ltd. is a fast growing IT industry and wants to implement a function to calculate the monthly income generated from all projects from their N no of clients like C1,C2,C3,C4....CN. The team wants to compare the time/steps required to execute this function on various inputs and analyse the complexity of each combination. Also draw a comparative chart. In each of the following functions N will be passed by user. Design the algorithm for the same and implement using the programming language of your choice. Make comparative analysis for various use cases &amp; input size.

    1. To calculate the sum of 1 to N number using loop.
    2. To calculate the sum of 1 to N number using the equation.
    3. To calculate sum of 1 to N numbers using recursion.

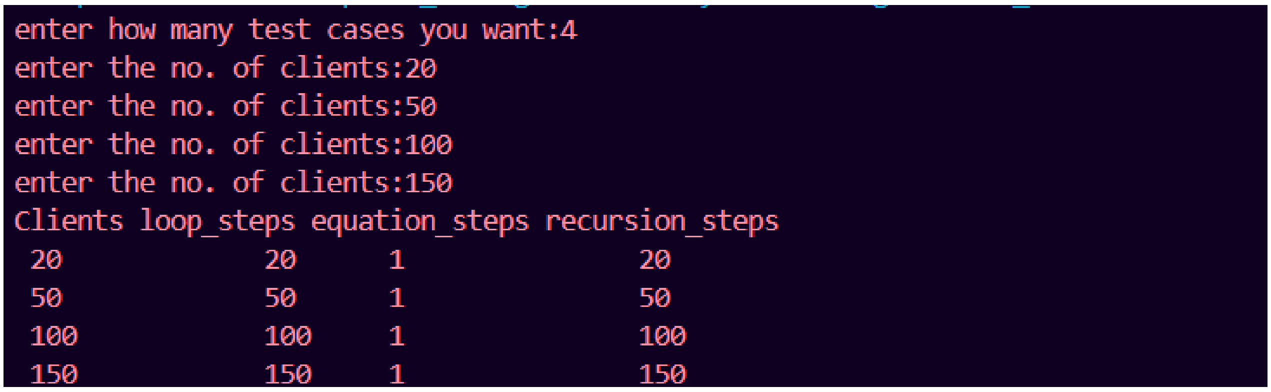
(2) Suppose a newly-born pair of rabbits, one male, one female, are put in a field. Rabbits are able to mate at the age of one month so that at the end of its second month a female can produce another pair of rabbits. Suppose that our rabbits never die and that the female always produces one new pair (one male, one female) every month from the second month on. How many pairs will there be in one year? Apply appropriate algorithm/method to find out the above problem and also solve them using iteration and recursive method. Compare the performance of two methods by counting the number of steps executed on various inputs. Also draw a comparative chart. Design the algorithm for the same and implement using the programming language of your choice. Make comparative analysis for various use cases &amp; input size.

CODE_01:

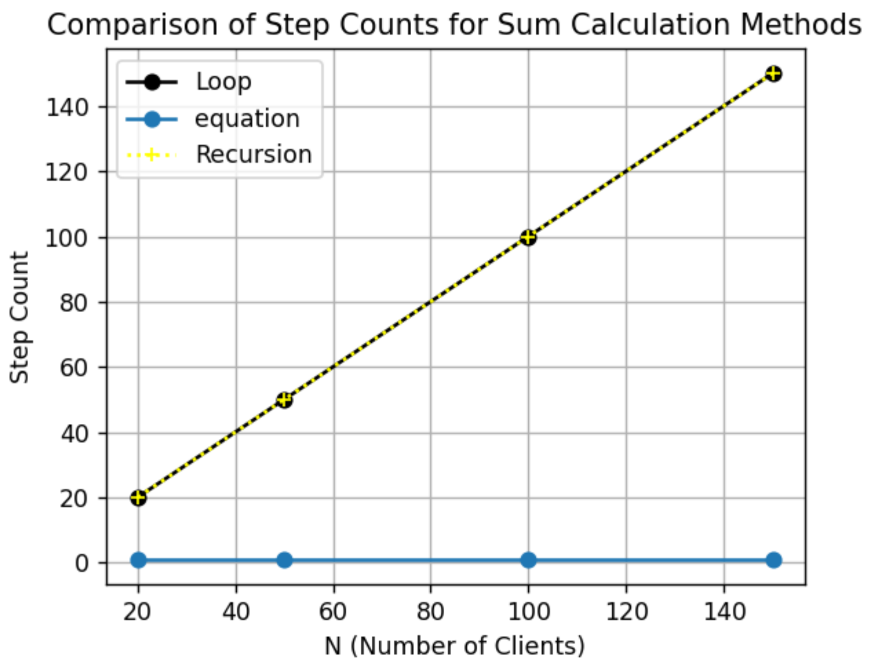


```python
import matplotlib.pyplot as plt
recursion_steps = 0
def sum_loop(n):
    steps = 0
    total = 0
    for i in range(1, n + 1):
        total = total + i
        steps = steps + 1
    return total, steps
def sum_equation(n):
    steps = 1
    total = n * (n + 1) // 2
    return total, steps
def sum_recursive(n):
    global recursion_steps
    recursion_steps = recursion_steps + 1
    if n == 1:
        return 1
    return n + sum_recursive(n-1)
n_values=[]
N=int(input("enter how many test cases you want:"))
for i in range(1,N+1):
    n_values.append(int(input("enter the no. of clients:")))
loop_step_list = []
equan_steps = []
recur_steps = []
print("Clients", "loop_steps", "equation_steps", "recursion_steps")
for n in n_values:
    result_loop, loop_steps = sum_loop(n)
    result_equation, equation_steps = sum_equation(n)
    recursion_steps = 0
    result_rec = sum_recursive(n)
    loop_step_list.append(loop_steps)
    equan_steps.append(equation_steps)
    recur_steps.append(recursion_steps)
    print(f" {n}      \t{loop_steps}  \t{equation_steps}\t\t{recursion_steps}")
plt.plot(n_values, loop_step_list, label="Loop", marker='o',color='black')
plt.plot(n_values, equan_steps, label="equation", marker='o')
plt.plot(n_values, recur_steps, label="Recursion", marker='+',linestyle='dotted',color='yellow')
plt.xlabel("N (Number of Clients)")
plt.ylabel("Step Count")
plt.title("Comparison of Step Counts for Sum Calculation Methods")
plt.legend()
plt.grid(True)
plt.show()
```

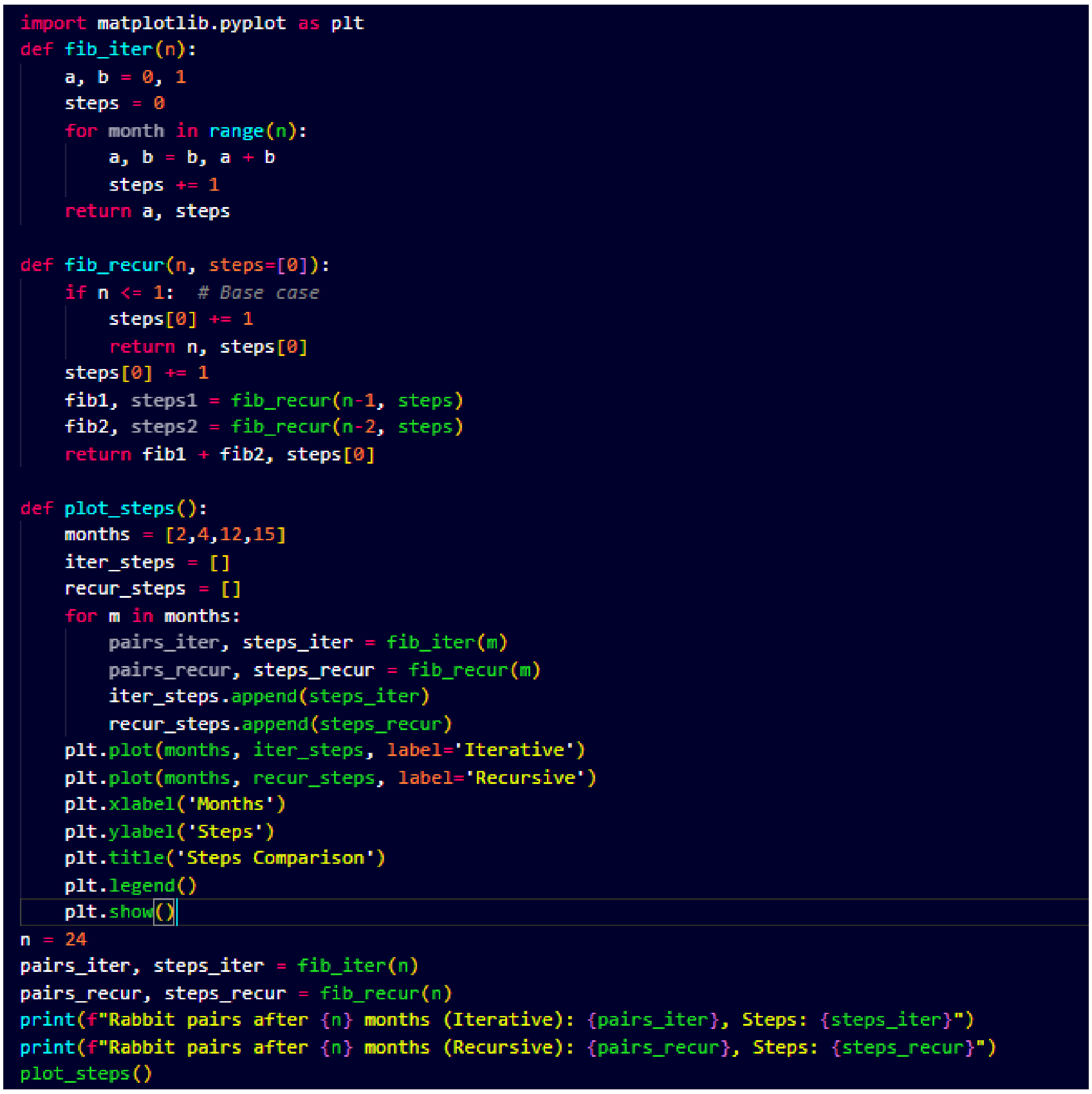
OUTPUT_01:



```
enter how many test cases you want:4
enter the no. of clients:20
enter the no. of clients:50
enter the no. of clients:100
enter the no. of clients:150
Clients loop_steps equation_steps recursion_steps
 20            20     1            20
 50            50     1            50
 100           100    1            100
 150           150    1            150
```

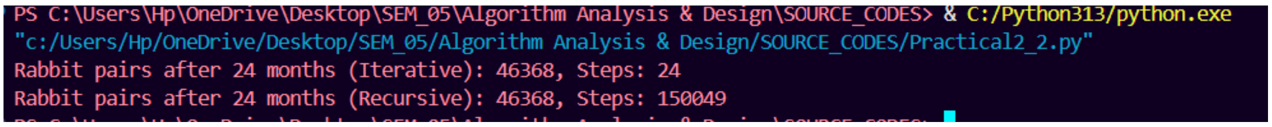


CODE_02:

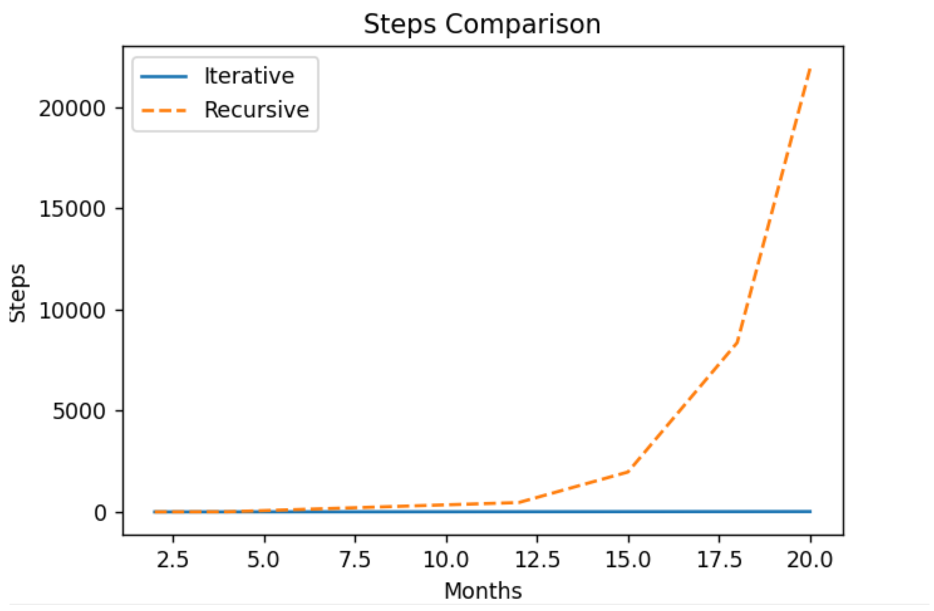


```
import matplotlib.pyplot as plt
def fib_iter(n):
    a, b = 0, 1
    steps = 0
    for month in range(n):
        a, b = b, a + b
        steps += 1
    return a, steps

def fib_recur(n, steps=[0]):
    if n <= 1:  # Base case
        steps[0] += 1
        return n, steps[0]
    steps[0] += 1
    fib1, steps1 = fib_recur(n-1, steps)
    fib2, steps2 = fib_recur(n-2, steps)
    return fib1 + fib2, steps[0]

def plot_steps():
    months = [2,4,12,15]
    iter_steps = []
    recur_steps = []
    for m in months:
        pairs_iter, steps_iter = fib_iter(m)
        pairs_recur, steps_recur = fib_recur(m)
        iter_steps.append(steps_iter)
        recur_steps.append(steps_recur)
    plt.plot(months, iter_steps, label='Iterative')
    plt.plot(months, recur_steps, label='Recursive')
    plt.xlabel('Months')
    plt.ylabel('Steps')
    plt.title('Steps Comparison')
    plt.legend()
    plt.show()
n = 24
pairs_iter, steps_iter = fib_iter(n)
pairs_recur, steps_recur = fib_recur(n)
print(f"Rabbit pairs after {n} months (Iterative): {pairs_iter}, Steps: {steps_iter}")
print(f"Rabbit pairs after {n} months (Recursive): {pairs_recur}, Steps: {steps_recur}")
plot_steps()
```

## OUTPUT_02:



```
PS C:\Users\Hp\OneDrive\Desktop\SEM_05\Algorithm Analysis & Design\SOURCE_CODES> & C:/Python313/python.exe
"c:/Users/Hp/OneDrive/Desktop/SEM_05/Algorithm Analysis & Design/SOURCE_CODES/Practical2_2.py"
Rabbit pairs after 24 months (Iterative): 46368, Steps: 24
Rabbit pairs after 24 months (Recursive): 46368, Steps: 150049
```


Steps Comparison
Iterative
Recursive
Steps
Months