

Hall Ticket No. : 2403A51L40

Name: Akhila G

Assignment 10.2

TASK 1:

Prompt: correct all syntax and logical errors.

Input:

```
def calculate_total(nums)
    sum = 0
    for n in nums
    sum += n
    return total
```

Output:

```
def calculate_total(nums):
    total = 0
    for n in nums:
        total += n
    return total
```

Explanation:

1. Added a colon (:) at the end of the function definition line.
2. Changed the variable name from 'sum' to 'total' to avoid conflict with the built-in sum() function.
3. Added a colon (:) at the end of the for loop line.

TASK 2:

Prompt: comply with standard coding style guidelines.abs

Input:

```
def findSum(a,b):return a+b
print(findSum(5,10))
```

Output:

```
def find_sum(a, b):
    return a + b
print(find_sum(5, 10))
```

Explanation:

1. Changed the function name from 'findSum' to 'find_sum' to follow the snake_case naming convention.
2. Added spaces around the operators for better readability.

TASK 3:

Prompt : improve code readability without changing its functionality.

Input:

```
def f(x,y):
    return x-y*2
print(f(10,3))
```

Output:

```
def calculate_difference(x, y):
    return x - y * 2
print(calculate_difference(10, 3))
```

Explanation:

1. Changed the function name from 'f' to 'calculate_difference' to make it more descriptive of its purpose.
2. Added indentation to the function body for better readability.

TASK 4:

Prompt: refactor repetitive code into reusable functions.

Input:

```
print("Hello Ram")
print("Hello Sita")
print("Hello Ravi")
```

Output:

```
def greet(name):
    print(f"Hello {name}")
greet("Ram")
greet("Sita")
greet("Ravi")
```

Explanation:

1. Created a function named 'greet' that takes a name as an argument and prints a greeting message.
2. Replaced the repetitive print statements with calls to the 'greet' function, passing the respective names as arguments.

TASK 5:

Prompt: optimize Python code for better performance.

Input:

```
numbers = [ ]  
for i in range(1, 500000):  
    numbers.append(i * i)  
print(len(numbers))
```

Output:

```
numbers = [i * i for i in range(1, 500000)]  
print(len(numbers))
```

Explanation:

1. Used a list comprehension to generate the list of squares instead of using a for loop and the append() method, which is more efficient and faster.
2. Added proper indentation to the code for better readability.