Ishita Haridasmani
14-58.

EXPERIMENT NO. 6

Aim: Write a program to implement code optimization

Theory:

Code optimization involves various techniques aimed at improving the efficiency of code, typically by reducing its execution time, memory usage or both. One of the fundamental principles of code optimization is to maintain the correctness of the program while enhancing it's performance. One commonly used optimization technique is loop optimization, which targets loop within a program since loop often represent a significant portion of execution time.

Consider the following python code that calculates the sum of squares

```
def sum_of_squares(lst):
    total = 0
    for num in lst:
        total += num ** 2
    return total
my_list = [1, 2, 3, 4, 5]
result = sum_of_squares(my_list)
print("Sum of squares:", result)
```

This code is straightforward & correct, but we can optimize it by using a mathematical formula for the sum of squares of consecutive numbers:

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

$1^2 + 2^2 + 3^2 + \dots + n^2 = n(n+1)(2n+1)/6$

The optimized version of code using this formula:

```
def sum_of_squares(lst):
    n = len(lst)
    return n*(n+1)*(2*n+1)//6

my_list = [1, 2, 3, 4, 5]
result = sum_of_squares(my_list)
print("Sum of squares:", result)
```

In this optimized version:

- We eliminated the loop entirely.
- We calculate the length of the list ('n') once.
- We use the formula directly to compute the sum of squares.

The optimization reduces both time complexity from $O(n) \rightarrow O(1)$ & space complexity (no need for loop variable 'num' & 'total').