Assignment 1

1. What is an algorithm and what are its essential properties?
2. Explain why analysis of algorithms is important? Explain: Worst Case, Best Case and Average Case Complexity with suitable example.
3. Write a function that takes an array of integers and returns an array of the squares of each number. What is the Big O notation for your function?
4. Write a function that takes a list of student names and scores and prints names of students who scored above the average score. Analyze the time complexity of your function. Provide the Big O, Big Omega, and Big Theta notations and justify each.
5. Consider a recursive function that computes the nth term of a mathematical series:

```
def series_term(n):
    if n < 2:
        return 1
    else:
        return series_term(n - 1) + series_term(n - 2)
```

   Determine the Big O notation for this function. Explain why this recursive approach has its particular complexity and discuss any potential improvements.

6. Given the following Python function:

```
def compute(n):
    sum = 0
    for i in range(n):
        for j in range(n):
            sum += i * j
    for k in range(n):
        sum += k
```

Analyze the time complexity of this function, providing the Big O and Big Theta notations. Explain the impact of each loop on the overall complexity.

7. Given the following functions, which statement is true regarding their asymptotic behavior?

- f1=2n^4 + 100n^3 + 50
- f2 = 3n^4

(A) f1=O(f2) and f2=O(f1) (B) f1=Θ(f2) (c) f1=o(f2) (D) f2=Ω(f1)

8. Solve the recurrence relation T(n) = 2T(n/2) + n using the substitution method.
9. Solve the recurrence relation T(n) = T(n/2) + 1 using the substitution method.
10. Solve the recurrence relation T(n)=T(n−1)+n using the substitution method
11. Solve T(n)=2T(n/4)+ √n using the substitution method.
12. Solve the recurrence relation T(n) = 3T(n/4) + n log n using the master method.
13. Solve T(n)=2T(n/2)+ √n using the Master Method.
14. Solve T(n)=4T(n/2)+ n^2logn using the Master Method.
15. Solve T(n)=2T(n/3)+n^2 using the Master Method
16. Solve T(n) = 4T(n/2) + n^2 log n Using the Recursion Tree Method
17. Solve T(n) = 2T(n/3) + n^2 Using the Recursion Tree Method
18. Given a list of numbers, write two different algorithms to sort the list: one using the bubble sort method and another using selection sort. Compare their performance by counting the number of operations each one performs.
19. Derive the best and worst-case time complexities of Insertion Sort. What role does the initial order of the input play in determining these complexities?
20. Determine the best and worst-case scenarios for swap operations in Selection Sort. Is there a difference in the number of swaps performed in these cases?