Question 1:

Imagine you are designing a file organization algorithm for a rapidly growing database. Each time you organize the files, you split the database into two halves and recursively organize each half. The cost of organizing a database is represented by the function T(n), where n is the number of files in the database. The recursive relationship is given by T(n) = 2 * T(n/2) + 1. Apply the substitution method to derive a closed-form expression for T(n). Show each step of your derivation and explain the logic behind your substitutions.

Input Format

- The input to this program is expected to be a single integer, num, which represents the size of the problem n.

Output Format

- The output of this program will be in the format T(num) = result, where num is the input integer, and result is the computed value of the recurrence relation for that n.

Title for Question 1: Recurrence Relation

Solution:

```
#include<iostream>
using namespace std;
// Function to solve the recurrence relation T(n) = 2 * T(n/2) + 1
int solveRecurrence(int n) {
    if (n == 0) {
        return 0;
    }
    return 2 * solveRecurrence(n / 2) + 1;
}

int main() {
    int num,result;
    cin>>num;
    result = solveRecurrence(num);
    cout<<"T("<<num<<") = "<<result;
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	8	T(8) = 15
2	4	T(4) = 7
3	9	T(9) = 15

S.No	Inputs	Outputs
4	13	T(13) = 15
5	12	T(12) = 15
6	2	T(2) = 3

White List:

Black List:

Question 2:

Imagine you are working on a project where you need to efficiently find the maximum element in a large array. Your team is considering a divide and conquer approach to optimize the algorithm. Describe how you would design the algorithm using a divide and conquer strategy, and then analyze its time complexity. Consider factors such as the size of the input array, the number of recursive calls, and the overall efficiency of the algorithm. Discuss potential advantages and limitations of this approach in comparison to other methods for finding the maximum element in an array.

Input Format

- The input to the program starts with a single integer, n, representing the number of elements in the array.
- This is followed by n integers, which are the elements of the array. The elements are provided in a single line separated by spaces or each on a new line.

Output Format

- The output of the program is a single line stating the maximum element in the array, prefixed with a message.

Title for Question 2: Maximum Element in an array

Solution:

```
#include <iostream>
using namespace std;

int findMax(const int arr[], int first, int last) {
    int mid,leftMax,rightMax;
    if (first == last) {
        return arr[first];
    }
    mid = (first + last) / 2;
    leftMax = findMax(arr, first, mid);
    rightMax = findMax(arr, mid + 1, last);
    return (leftMax > rightMax) ? leftMax : rightMax;
}
```

```
int main() {
    int ind,n,maxElement;
    cin>>n;
    int arr[n];
    for(ind=0;ind<n;ind++) {
        cin>>arr[ind];
    }
    maxElement = findMax(arr, 0, n - 1);
    cout<<"The maximum element in the array is: "<<maxElement;
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	5 3 7 1 9 4	The maximum element in the array is: 9
2	7 -10 -20 -30 -40 -50 -60 -70	The maximum element in the array is: -10
3	4 10 10 10 10	The maximum element in the array is: 10
4	1 100	The maximum element in the array is: 100
5	3 -5 0 -2	The maximum element in the array is: 0
6	8 12 45 23 6 78 34 67 19	The maximum element in the array is: 78

White List:

Black List:

Question 3:

Imagine you are working on a project that involves processing a large dataset of financial transactions. Each transaction has a unique identifier, and you need to calculate the value of the nth term in an arithmetic progression series based on these identifiers. The arithmetic progression is defined as follows: starting from the first transaction with an identifier 'a,' each subsequent transaction's identifier is obtained by adding a constant difference 'd' to the previous identifier. Now, considering that you have implemented an algorithm to calculate the nth term of this arithmetic progression series, discuss how you would approach analyzing its time complexity using the master theorem.

Input Format

- The input to this program should consist of three integers separated by spaces:
- The first term of the arithmetic progression.
- The common difference between any two consecutive terms of the arithmetic progression.
- The position (n) of the term in the arithmetic progression that you want to find.

Output Format

- The output will be a single line that states the nth term of the arithmetic progression, where n is the position specified in the input.

Title for Question 3: Nth term of an Arithmetic Progression

Solution:

```
#include <iostream>
using namespace std;

int nthTermOfAP(int a1, int d, int n) {
    return a1 + (n - 1) * d;
}

int main() {
    int firstTerm,difference,pos,nthTerm;
    cin>>firstTerm>>difference>>pos;
    nthTerm = nthTermOfAP(firstTerm, difference, pos);
    cout <<"The "<<pos<<"th term of the arithmetic progression is: "<<nth
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	1 2 5	The 5th term of the arithmetic progression is: 9
2	-2 4 7	The 7th term of the arithmetic progression is: 22
3	10 5 2	The 2th term of the arithmetic progression is: 15
4	100 1 10	The 10th term of the arithmetic progression is: 109
5	0 3 3	The 3th term of the arithmetic progression is: 6
6	2 3 5	The 5th term of the arithmetic progression is: 14

White List:

Black List:

Question 4:

Imagine you are working on a project that involves analyzing stock prices. You have a dataset representing the daily stock prices of a company over a certain period. The goal is to find the maximum subarray sum, which corresponds to the maximum profit that could have been obtained by buying and selling the company's stock on different days.

Input format

- The first line of input should be a single integer, n, representing the number of elements in the array.

- The second line of input should consist of n integers separated by spaces, representing the elements of the array.

Output format

- The output of the program is a single line that states the maximum sum of any subarray within the given array.

Title for Question 4: Maximum Subarray Sum

Solution:

```
#include <iostream>
#include <vector>
#define Max(a,b) (a>b)?a:b
using namespace std;

int maximum(int a, int b, int c) {
    return Max(Max(a,b), c);
}

int maxCrossingSum(int arr[], int low, int mid, int high) {
    int leftSum = 0;
    int sum = 0;

    // Find the maximum sum in the left subarray
    for (int i = mid; i >= low; --i) {
        sum += arr[i];
        leftSum = Max(leftSum, sum);
    }

    int rightSum = 0;
    sum = 0;

    // Find the maximum sum in the right subarray
    for (int i = mid + 1; i <= high; ++i) {
        sum += arr[i];
        rightSum = Max(rightSum, sum);
    }

    // Return the sum of the left and right subarrays
    return leftSum + rightSum;
}

int maxSubarraySum(int arr[], int low, int high) {
    if (low == high) {
        return arr[low];
    }
    int mid = (low + high) / 2;
    // Recursively find the maximum subarray sum in the left and right su
    int leftSum = maxSubarraySum(arr, low, mid);
    int rightSum = maxSubarraySum(arr, mid + 1, high);
    int crossSum = maxCrossingSum(arr, low, mid, high);
```

```
    return maximum(leftSum, rightSum, crossSum);
}

int main() {
    int n,ind,maxSum;
    cin>>n;
    int arr[n];
    for(ind=0;ind<n;ind++) {
        cin>>arr[ind];
    }
    maxSum = maxSubarraySum(arr, 0, n - 1);
    cout << "Maximum subarray sum: " << maxSum;
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	8 -2 -3 4 -1 -2 1 5 -3	Maximum subarray sum: 7
2	5 1 2 3 4 5	Maximum subarray sum: 15
3	6 -1 -2 -3 -4 -5 -6	Maximum subarray sum: -1
4	3 -2 5 -1	Maximum subarray sum: 5
5	4 -2 1 -3 4	Maximum subarray sum: 4
6	5 -10 -5 -2 -1 -7	Maximum subarray sum: 0

White List:

Black List:

Question 5:

The Prisoners are arranged in a circle, and every kth person will be spared, with counting starting from the first person. If they refuse to play or are chosen to be executed, their fate is sealed.

Input Format

- n: The total number of people in the circle.
- k: The step count after which a person is executed. That is, k-1 people are skipped, and the kth person is executed.

Output Format

- The output of the program is a single line stating the position of the survivor, i.e., the position in the initial circle of the person who will remain alive at the end.

Title for Question 5: Josephus Problem

Solution:

```
#include<iostream>
using namespace std;

int josephus(int n, int k) {
    if (n == 1)
        return 1;
    return (josephus(n - 1, k) + k - 1) % n + 1;
}

int main() {
    int n, k, survivor;
    cin>>n>>k;
    survivor = josephus(n, k);
    cout << "The survivor is at position: " << survivor;
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	5 2	The survivor is at position: 3
2	10 3	The survivor is at position: 4
3	9 4	The survivor is at position: 1
4	7 1	The survivor is at position: 7
5	12 3	The survivor is at position: 10
6	6 3	The survivor is at position: 1

White List:

Black List:

Question 6:

Imagine you are developing a program to calculate the nth term of the Tribonacci sequence using a recursive function with the recurrence relation T(n) = T(n-1) + T(n-2) + T(n-3). A user is interested in finding the Nth term of the Tribonacci sequence. How would you implement and call the recursive function to provide the user with the correct result.

Input Format

The input format for this program is a single integer, n, which represents the term of the Tribonacci sequence that you want to calculate.

Output Format

The output of the program is a single line that provides the nth term of the Tribonacci sequence.

Title for Question 6: Tribonacci Sequence

Solution:

```
#include<iostream>
using namespace std;

int tribonacci(int n) {
    if (n == 0)
        return 0;
    if (n == 1 || n == 2)
        return 1;
    return tribonacci(n - 1) + tribonacci(n - 2) + tribonacci(n - 3);
}

int main() {
    int n,result;
    cin >> n;
    result = tribonacci(n);
    cout << "The " << n << "th term of the Tribonacci sequence is: " << r
    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	1	The 1th term of the Tribonacci sequence is: 1
2	5	The 5th term of the Tribonacci sequence is: 7
3	9	The 9th term of the Tribonacci sequence is: 81
4	3	The 3th term of the Tribonacci sequence is: 2
5	4	The 4th term of the Tribonacci sequence is: 4
6	0	The 0th term of the Tribonacci sequence is: 0

White List:

Black List:
