Question 1:

Write a program that adds and subtracts two integer values using binary C++
Operator Overloading:

????**Input Format:**

- The first line contains an integer representing the first number.
- The second line contains an integer representing the second number.

Output Format:

- The first line contains the phrase "Sum = " followed by the sum of the two input numbers.
- The second line contains the phrase "Difference = " followed by the difference of the two input numbers (i.e., the result of subtracting the second number from the first number).

Title for Question 1: Operator Overloading- Sum difference

Solution:

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

class Number {
    int value;

public:
    // Default constructor
    Number() : value(0) {}

    // Parameterized constructor
    Number(int v) : value(v) {}

    // Overload + operator to add two numbers
    Number operator+(const Number& n) {
        return Number(value + n.value);
    }

    // Overload - operator to subtract two numbers
    Number operator-(const Number& n) {
        return Number(value - n.value);
    }

    // Overload << operator to output number value
    friend ostream& operator<<(ostream& out, const Number& n) {
        out << n.value;
        return out;
    }

    // Overload >> operator to input number value
    friend istream& operator>>(istream& in, Number& n) {
        in >> n.value;
        return in;
    }
};
int main() {
    Number num1, num2, sum, diff;

   // cout << "Enter first integer: ";
    cin >> num1;

    //cout << "Enter second integer: ";
    cin >> num2;

    sum = num1 + num2;
    diff = num1 - num2;

    cout << "Sum = " << sum << endl;
    cout << "Difference = " << diff << endl;

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	5 3	Sum = 8 Difference = 2
2	987 64	Sum = 1051 Difference = 923
3	34 -56	Sum = -22 Difference = 90
4	-9 -7	Sum = -16 Difference = -2
5	-43 86	Sum = 43 Difference = -129
6	987654 23456789	Sum = 24444443 Difference = -22469135

White List:

Black List:

Question 2:

Create two classes:

Cuboid

The Cuboid class should have three data fields- length, width and height of int types. The class should have display() method, to print the length, width and height of the cuboid separated by space.

CuboidVol

The CuboidVol class is derived from Cuboid class, i.e., it is the sub-class of Cuboid class. The class should have read_input() method, to read the values of length, width and height of the Cuboid. The CuboidVol class should also overload the display() method to print the volume of the Cuboid (length * width * height).

Input Format:

- The first line contains the number of test cases and one and only line of each test case contains 3 space separated integer denoting length, width, and height of the Cuboid

Output Format:

- The output should consist of exactly two lines:
- In the first line, print the length, width, and height of the cuboid separated by space.
- In the second line, print the volume of the cuboid.

Constraints:

0 <= (length, width, height) <= 100

Title for Question 2: Single inheritance- Cuboid Volume

Solution:

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

// Base class: Cuboid
class Cuboid {
protected:
    int length, width, height;

public:
    Cuboid() : length(0), width(0), height(0) {}

    virtual void display() {
        cout << length << " " << width << " " << height << endl;
    }
};
```

```
// Derived class: CuboidVol
class CuboidVol : public Cuboid {
public:
    void read_input() {
        cin >> length >> width >> height;
    }

    void display() override {
        Cuboid::display();  // Call base class display to print length, w
        cout << length * width * height << endl;  // Print the volume
    }
};

int main() {
    int t;
    //cout << "Enter number of test cases: ";
    cin >> t;

    while(t--) {
        CuboidVol cube;
        //cout << "Enter length, width and height separated by space: ";
        cube.read_input();
        cube.display();
    }

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	2 3 4 5 2 2 2	3 4 5 60 2 2 2 8
2	1 2 3 4	2 3 4 24
3	2 5 5 5 3 4 7	5 5 5 125 3 4 7 84
4	3 1 2 3 10 10 10 7 8 9	1 2 3 6 10 10 10 1000 7 8 9 504
5	2 6 6 6 4 5 6	6 6 6 216 4 5 6 120
6	1 12 8 5	12 8 5 480

White List:

Black List:

Question 3:

The existing code does not handle the case where either x or y is zero or negative. Modify the code Write a C++ program that calculates the GCD and LCM of a series of pairs of integers. The program should continuously accept user inputs for pairs of integers until the user decides to stop. After each calculation, the program should ask the user if they want to perform another calculation and continue if the user enters 'Y' or 'y'. The program should display the GCD and LCM for each pair of integers.

Input Format:

- The user is prompted to enter two integer numbers separated by a space or on new lines for each pair of integers.
- After each calculation, the program asks if the user wants to continue ('Y' or 'y' to continue, any other input to stop).

Output Format:

- For each pair of integers, display the GCD and LCM.
- If the user chooses to continue, repeat the process for the next pair of integers.

Title for Question 3: LCM and GCD -Single Inheritance

Solution:

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

// Base class: Numbers
class Numbers {
protected:
    int num1, num2;
public:
    // Constructor to initialize numbers
    Numbers(int x, int y) : num1(x), num2(y) {}
};

// Derived class: Calculator
class Calculator : public Numbers {
public:
    // Constructor to initialize numbers using base class constructor
    Calculator(int x, int y) : Numbers(x, y) {}

    // Function to compute GCD
    int gcd() {
        int a = num1, b = num2;
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }

    // Function to compute LCM
    int lcm() {
        int gcdValue = gcd();
        return (num1 * num2) / gcdValue;
    }
};

int main() {
    int x, y;
    //cout << "Enter two numbers:\n";
    cin >> x >> y;

    Calculator calc(x, y);
    cout << "GCD of " << x << " and " << y << " is: " << calc.gcd() << en
    cout << "LCM of " << x << " and " << y << " is: " << calc.lcm() << en

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	12 15	GCD of 12 and 15 is: 3 LCM of 12 and 15 is: 60
2	17 23	GCD of 17 and 23 is: 1 LCM of 17 and 23 is: 391
3	12 18	GCD of 12 and 18 is: 6 LCM of 12 and 18 is: 36
4	60 40	GCD of 60 and 40 is: 20 LCM of 60 and 40 is: 120
5	27 -72	GCD of 27 and -72 is: 9 LCM of 27 and -72 is: -216
6	98765 123	GCD of 98765 and 123 is: 1 LCM of 98765 and 123 is: 12148095

White List:

Black List:

Question 4:

Create a C++ program modeling a real estate system for house details. Implement a base class for land type, a derived class for land dimensions, and another base class for the number of floors. Finally, derive a class for the house to calculate and display the total area required.

Input Format:

- Enter land type (Residential/Commercial): [Land Type]
- Enter land width (in meters): [Width]
- Enter land length (in meters): [Length]
- Enter number of floors: [Number of Floors]

Output Format:

The Output Consist of

- The first line output represent Land Type: [Land Type]
- The Second line output represent Number of Floors: [Number of Floors]
- The Third line output represent Area per Floor: [Area per Floor] square meters
- Total Area required: [Total Area] square meters

Title for Question 4: Build a house -Hybrid Inheritance

Solution:

```
#include <iostream>
using namespace std;
const int MAX_SIZE = 100;
// Base class: Land
class Land {
protected:
    char type[MAX_SIZE];
public:
    void getType() {
        //cout << "Enter land type (Residential/Commercial): ";
        cin >> type;
    }
};

// Derived class: Dimensions from Land
class Dimensions : public Land {
protected:
    float width, length;
public:
    void getDimensions() {
        //cout << "Enter land width (in meters): ";
        cin >> width;
        //cout << "Enter land length (in meters): ";
        cin >> length;
    }
    float area() {
        return width * length;
    }
};

// Another Base class: Floor
class Floor {
protected:
    int floors;
public:
    void getFloors() {
        //cout << "Enter number of floors: ";
        cin >> floors;
    }
};

// Final Derived class: House from Dimensions and Floor
class House : public Dimensions, public Floor {
public:
    void displayArea() {
        float totalArea = area() * floors;
     //   cout << "--- House Details ---\n";
        cout << "Land Type: " << type << endl;
        cout << "Number of Floors: " << floors << endl;
        cout << "Area per Floor: " << area() << " square meters" << endl;
        cout << "Total Area required: " << totalArea << " square meters"
    }
};

int main() {
    House h;
    h.getType();
    h.getDimensions();
    h.getFloors();
    h.displayArea();

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	Residential 10 20 3	Land Type: Residential Number of Floors: 3 Area per Floor: 200 square meters Total Area required: 600 square meters
2	Commercial 190 70 25	Land Type: Commercial Number of Floors: 25 Area per Floor: 13300 square meters Total Area required: 332500 square meters
3	Residential 45 98 6	Land Type: Residential Number of Floors: 6 Area per Floor: 4410 square meters Total Area required: 26460 square meters
4	Commercial 65 56 16	Land Type: Commercial Number of Floors: 16 Area per Floor: 3640 square meters Total Area required: 58240 square meters
5	Residential 25 4 3	Land Type: Residential Number of Floors: 3 Area per Floor: 100 square meters Total Area required: 300 square meters
6	Commercial 178 98 14	Land Type: Commercial Number of Floors: 14 Area per Floor: 17444 square meters Total Area required: 244216 square meters

White List:

Black List:

Question 5:

Write a program that constructs a family tree by taking the names of a grandparent, parent, and child as input. The program should display the constructor calls for each family member in the correct order. After constructing the family tree, it should display "Family tree constructed."

Input Format:

- The user is prompted to enter the name of the grandparent, parent, and child (strings).
- After entering the names, the program constructs the family tree.

Output Format:

- The program starts by displaying "Constructing the family tree..." followed by a series of constructor calls indicating which constructor was called for each family member.
- Finally, it displays "Family tree constructed."

Title for Question 5: Family tree - Multilevel inheritance.

Solution:

```
#include <iostream>
#include <string>

class Grandparent {
public:
    std::string grandparentName;

    Grandparent(std::string name) : grandparentName(name) {
        std::cout << "Grandparent's constructor called for " << grandpare
    }
};

class Parent : public Grandparent {
public:
    std::string parentName;

    Parent(std::string grandparentName, std::string name) : Grandparent(g
        std::cout << "Parent's constructor called for " << parentName << 
    }
};

class Child : public Parent {
public:
    std::string childName;

    Child(std::string grandparentName, std::string parentName, std::strin
        : Parent(grandparentName, parentName), childName(name) {
        std::cout << "Child's constructor called for " << childName << '\
    }
};

int main() {
    std::string grandparentName, parentName, childName;

   // std::cout << "Enter the name of the grandparent: ";
    std::getline(std::cin, grandparentName);

   // std::cout << "Enter the name of the parent: ";
    std::getline(std::cin, parentName);

   // std::cout << "Enter the name of the child: ";
    std::getline(std::cin, childName);

    std::cout << "Constructing the family tree...\n";
    Child familyMember(grandparentName, parentName, childName);
    std::cout << "Family tree constructed.";

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs

1	Ram,Meena Dinesh,Keerthi Chandru	Constructing the family tree... Grandparent's constructor called for Ram,Meena Parent's constructor called for Dinesh,Keerthi Child's constructor called for Chandru Family tree constructed.
2	Alice Bob Carol Doug Emily	Constructing the family tree... Grandparent's constructor called for Alice Parent's constructor called for Bob Child's constructor called for Carol Doug Emily Family tree constructed.
3	Elizabeth Ann Johnson Robert William Smith Jennifer Lynn Davis	Constructing the family tree... Grandparent's constructor called for Elizabeth Ann Johnson Parent's constructor called for Robert William Smith Child's constructor called for Jennifer Lynn Davis Family tree constructed.
4	Johnson Davis	Constructing the family tree... Grandparent's constructor called for Johnson Parent's constructor called for Davis Child's constructor called for Family tree constructed.
5	Sam Anto Yamini	Constructing the family tree... Grandparent's constructor called for Sam Parent's constructor called for Anto Child's constructor called for Yamini Family tree constructed.
6	Restro Uncle Manoj	Constructing the family tree... Grandparent's constructor called for Restro Parent's constructor called for Uncle Child's constructor called for Manoj Family tree constructed.

White List:

Black List:

Question 6:

You are working with a class named ArrayGroup that is designed to group numbers from an array based on a specified size parameter. The private data member of the class stores these grouped numbers. There is a friend function named fillGroup that fills this data member by grouping numbers from a given array based on the size parameter.

Write a program that uses the ArrayGroup class and the fillGroup friend function to solve the problem described above.

Input Format

- An integer n (1 ? n ? 10^5), representing the number of elements in the array.
- Next line contains n space-separated integers.
- An integer size (1 ? size ? n) - the maximum size for each sub-array.

Output Format:

- Multiple lines where each line contains space-separated integers representing a group.
- The groups should be displayed in the order they were formed.

Title for Question 6: Group in order // friend function

Solution:

```
#include <iostream>

class ArrayGroup {
private:
    int **groupedNumbers;
    int groupCount;
    int groupSize;

public:
    ArrayGroup() {
        groupedNumbers = nullptr;
        groupCount = 0;
        groupSize = 0;
    }

    ~ArrayGroup() {
        if (groupedNumbers != nullptr) {
            for (int i = 0; i < groupCount; ++i) {
                delete[] groupedNumbers[i];
            }
            delete[] groupedNumbers;
        }
    }

    friend void fillGroup(ArrayGroup &group, const int *arr, int n, int s

    void displayGroups() {
        for (int i = 0; i < groupCount; ++i) {
            for (int j = 0; j < groupSize; ++j) {
                if (groupedNumbers[i][j] == 0) {
                    std::cout << " ";
                } else {
                    std::cout << groupedNumbers[i][j] << " ";
                }
            }
            std::cout << std::endl;
        }
    }
};

void fillGroup(ArrayGroup &group, const int *arr, int n, int size) {
    group.groupCount = (n + size - 1) / size;
    group.groupSize = size;
    group.groupedNumbers = new int *[group.groupCount];

    int index = 0;

    for (int i = 0; i < group.groupCount; ++i) {
        group.groupedNumbers[i] = new int[size];
        for (int j = 0; j < size; ++j) {
            if (index < n) {
                group.groupedNumbers[i][j] = arr[index];
                ++index;
            } else {
                group.groupedNumbers[i][j] = 0; // Set unused values to 0
            }
        }
    }
}

int main() {
    int n;
    std::cin >> n;

    int *arr = new int[n];
    for (int i = 0; i < n; ++i) {
        std::cin >> arr[i];
    }

    int size;
    std::cin >> size;

    ArrayGroup group;
    fillGroup(group, arr, n, size);

    group.displayGroups();

    delete[] arr;

    return 0;
}
```

TestCases:

S.No	Inputs	Outputs
1	6 10 20 30 40 50 60 4	10 20 30 40 50 60
2	4 1 2 3 4 2	1 2 3 4
3	7 1 2 3 4 5 6 7 3	1 2 3 4 5 6 7
4	9 1 2 3 4 5 16 17 18 19 3	1 2 3 4 5 16 17 18 19
5	2 1 2 1	1 2
6	5 5 -4 -3 2 1 2	5 -4 -3 2 1

White List:

Black List: