Question01.cpp

```cpp
// DSA LAB 7
// <---Q1--->
// Write a program and recurrence relation to find the Fibonacci series of n where n>2.

#include <iostream>
using namespace std;

// Function to find the nth Fibonacci number using recursion
int fibonacciRecursive(int n) {
    if (n <= 1) {
        return n;
    } else {
        return fibonacciRecursive(n - 1) + fibonacciRecursive(n - 2);
    }
};

int main() {
    int n;

    // Input the value of n
    cout << "Enter the value of n (n > 2): ";
    cin >> n;

    // Check if n is greater than 2
    if (n <= 2) {
        cout << "Invalid input. Please enter n > 2." << endl;
        return 1; // Return an error code
    }

    // Using recursion to find the nth Fibonacci number
    cout << "Fibonacci series using recursion:" << endl;
    for (int i = 0; i < n; ++i) {
        cout << fibonacciRecursive(i) << " ";
    }
    cout << endl;

    return 0;
}

```

Question02.cpp

```cpp
// DSA LAB 7
// <---Q2--->

// Write a program and recurrence relation to find the Factorial of n where n>2.

#include <iostream>
using namespace std;

// Function to find the factorial of a number using recursion
unsigned long long factorialRecursive(int n)
{
    if (n <= 1)
    {
        return 1;
    }
    else
    {
        return n * factorialRecursive(n - 1);
    }
}

int main()
{
    int n;

    // Input the value of n
    cout << "Enter the value of n (n > 2): ";
    cin >> n;

    // Check if n is greater than 2
    if (n <= 2)
    {
        cout << "Invalid input. Please enter n > 2." << endl;
        return 1; // Return an error code
    }

    // Using recursion to find the factorial of n
    cout << "Factorial using recursion: " << factorialRecursive(n) << endl;
    return 0;
}

```

Question03.cpp

```cpp
// DSA LAB 7
// <---Q3--->

// Write a recursive function which will take input from the user until a special character
// (also selected by the user) is not entered. Then print all the input in reverse.
// Sample Input:
// Enter Special Character: !
// Enter Character: A
// Enter Character: B
// Enter Character: C
// Enter Character: !
// Sample Output: C B A


#include <iostream>
#include <string>

using namespace std;

// Recursive function to read input until a special character is encountered
void readInputAndReverse(char specialCharacter)
{
    char ch;
    cout << "Enter Character: ";
    cin >> ch;

    // Check if the entered character is the special character
    if (ch == specialCharacter)
    {
        // Base case: Stop recursion
        return;
    }

    // Recursive call for the next character
    readInputAndReverse(specialCharacter);

    // Print the entered character after the recursive call (post-order)
    cout << ch << " ";
}

int main()
{
    char specialCharacter;

    // Input the special character
    cout << "Enter Special Character: ";
    cin >> specialCharacter;
    cout << endl;

    // Read input until the special character is encountered and print in reverse
    readInputAndReverse(specialCharacter);
    cout << "Sample Output ";
    cout << endl;

    return 0;
}

```

Question04.cpp

```cpp
// DSA LAB 7
// <---Q4--->

// Write a recursive function which will raise a number (double) to a non-negative integer
// power n. The function receives the double value and integer as arguments.

#include <iostream>

using namespace std;

// Recursive function to calculate the power of a double value
double power(double base, int exponent) {
    // Base case: Any number raised to the power of 0 is 1
    if (exponent == 0) {
        return 1.0;
    }

    // Recursive case: Multiply the base by the result of the recursive call
    // with a reduced exponent
    return base * power(base, exponent - 1);
}

int main() {
    double base;
    int exponent;

    // Input the base and exponent
    cout << "Enter the base (double): ";
    cin >> base;

    cout << "Enter the exponent (non-negative integer): ";
    cin >> exponent;

    // Check if the exponent is non-negative
    if (exponent < 0) {
        cout << "Invalid exponent. Please enter a non-negative integer." << endl;
        return 1; // Return an error code
    }

    // Calculate and print the result
    double result = power(base, exponent);
    cout << base << " raised to the power " << exponent << " is: " << result << endl;

    return 0;
}

```

Question05.cpp

```cpp
// DSA LAB 7
// <---Q5--->

// Write a recursive method that for a positive integer n prints odd numbers

// a. between 1 and n
// b. between n and 1


#include <iostream>
using namespace std;
void printOddNumbersUpToN(int n) {
    if (n >= 1) {
        printOddNumbersUpToN(n - 1);
        if (n % 2 != 0) {
            cout << n << " ";
        }
    }
}
void printOddNumbersDownTo1(int n) {
    if (n >= 1) {
        if (n % 2 != 0) {
            cout << n << " ";
        }
        printOddNumbersDownTo1(n - 1);
    }
}
int main() {
    int n;
    do {
        cout << "Enter a positive integer (n): ";
        cin >> n;
    } while (n <= 0);
    cout << "a. Odd numbers between 1 and " << n << ": ";
    printOddNumbersUpToN(n);
    cout << "\nb. Odd numbers between " << n << " and 1: ";
    printOddNumbersDownTo1(n);
    cout << endl;
    return 0;
}
```