```cpp
// Question_1.cpp : This file contains the 'main' function. Program execution begins and ends there.
// /-----------------------------------------------------------------------------------------------------------------------
//Name                              Sai Chaitanya Kilambi
//Course                            CPSC 131 Data Structures, Fall, 2022
//Assignment                        No.8 question:1
//Due date                          10/26/2022
// Purpose:
// This program stores the data in ordered linked list and displays their union
//------------------------------------------------------------------------------------------------------------------------
// list of libraries
//
//importing the required libraries

#include <iostream>

using namespace std;

template <class T>
class ORDER
{
private:
    struct node
    {
        T info;
        node* next;
    };
    node* order;
public:
    ORDER() { order = NULL; }// constructore
    bool emptyOrder()
    {
        return (order == NULL) ? true : false;
    }
    void pushOrder(T x)//
    {
        //insert x in the list and keep the list sorted
        node* r = new node; r->info = x;
        r->next = NULL;
        //find the insertion place;
        node* p = order; node* q = order;
        if (order == NULL)
            order = r;
        else
        {
            while (p != NULL && x > p->info)
            {
                q = p; p = p->next;
            }
            if (p == q)
            { //insert in front
                r->next = p; order = r;
            }
            else
            { //insert at the rear
                r->next = p; q->next = r;
            }
        }
    }
    void displayOrder() {
        node* p = order;
        while (p != NULL)
        {
            cout << p->info << "-->"; p = p->next;
        }
        cout << "NULL\n";
    }
    T popOrder()
    {
        //return the info of the first node and then
        //delete that node
        T popedElement;
        node* p = order;
        popedElement = p->info;
        order = p->next;
        delete p;
        return popedElement;
    }
};


int main()
{
    //find the union of two sets A and B
    int A[4] = { 3,8,4,1 };
    int B[5] = { 5,8,6,4,7 };
    // insert elements of A in setA
    ORDER<int> setA; ORDER<int> setB;
    for (int i = 0; i < 4; ++i)
        setA.pushOrder(A[i]);
    //insert elements of B in setB
    for (int i = 0; i < 5; ++i)
        setB.pushOrder(B[i]);
    //display both sets
    cout << "Set A = "; setA.displayOrder();
    cout << "Set B = "; setB.displayOrder();
    //find AB, the union of A and B
    ORDER<int> setAB;
    int A_elt = setA.popOrder();
    int B_elt = setB.popOrder();
    while (!setA.emptyOrder() || !setB.emptyOrder())
    {
        if (A_elt == B_elt)
        {
            setAB.pushOrder(A_elt);//collect their common elements
            A_elt = setA.popOrder();// go to next elt of setA
        }
        else {
            if (A_elt < B_elt) {
                setAB.pushOrder(A_elt);
                A_elt = setA.popOrder();//look at the next element of setA

            }
            else {

                B_elt = setB.popOrder();//look at the next element of setB
                setAB.pushOrder(B_elt);
            }
        }
    }

    //display their union set, setAB
    cout << "A union B = "; setAB.displayOrder();
    return 0;
}



```