```
...ssignments\131_Assignment_8\Question_1\Question_1.cpp
 1 // Question_1.cpp : This file contains the 'main' function. Program
     execution begins and ends there.
 2 // /
 3 //Name
                                 Sai Chaitanya Kilambi
 4 //Course
                                CPSC 131 Data Structures, Fall, 2022
 5 //Assignment
                               No.8 question:1
 6 //Due date
                                10/26/2022
 7 // Purpose:
 8 // This program stores the data in ordered linked list and displays their >
     union
 9 //------
10 // list of libraries
11 //
12 //importing the required libraries
13
14 #include <iostream>
15
16 using namespace std;
17
18 template <class T>
19 class ORDER
20 {
21 private:
     struct node
22
23
24
          T info;
25
          node* next;
26
       };
27
       node* order;
28 public:
       ORDER() { order = NULL; }// constructore
29
30
       bool emptyOrder()
31
32
           return (order == NULL) ? true : false;
33
       void pushOrder(T x)//
34
35
           //insert x in the list and keep the list sorted
36
37
           node* r = new node; r->info = x;
38
           r->next = NULL;
39
           //find the insertion place;
40
           node* p = order; node* q = order;
           if (order == NULL)
41
42
              order = r;
43
           else
44
           {
```

```
...ssignments\131_Assignment_8\Question_1\Question_1.cpp
```

```
2
```

```
45
                while (p != NULL && x > p->info)
46
                {
47
                    q = p; p = p->next;
48
49
                if (p == q)
50
                { //insert in front
51
                    r->next = p; order = r;
52
                }
53
                else
54
                { //insert at the rear
55
                    r->next = p; q->next = r;
56
                }
57
            }
58
        }
        void displayOrder() {
59
60
            node* p = order;
            while (p != NULL)
61
62
63
                cout << p->info << "-->"; p = p->next;
64
65
            cout << "NULL\n";</pre>
66
        }
67
        T popOrder()
68
            //return the info of the first node and then
69
70
            //delete that node
71
            T popedElement;
72
            node* p = order;
            popedElement = p->info;
73
74
            order = p->next;
75
            delete p;
76
            return popedElement;
77
        }
78 };
79
80
81 int main()
82 {
83
        //find the union of two sets A and B
        int A[4] = \{ 3,8,4,1 \};
84
85
        int B[5] = \{ 5,8,6,4,7 \};
86
        // insert elements of A in setA
87
        ORDER<int> setA; ORDER<int> setB;
88
        for (int i = 0; i < 4; ++i)
89
            setA.pushOrder(A[i]);
90
        //insert elements of B in setB
91
        for (int i = 0; i < 5; ++i)
92
            setB.pushOrder(B[i]);
93
        //display both sets
```

```
... s signments \verb|\131_Assignment_8| Question_1\\ Question_1. cpp
```

```
3
```

```
cout << "Set A = "; setA.displayOrder();</pre>
         cout << "Set B = "; setB.displayOrder();</pre>
 95
 96
         //find AB, the union of A and B
 97
         ORDER<int> setAB;
         int A_elt = setA.popOrder();
 98
         int B_elt = setB.popOrder();
99
         while (!setA.emptyOrder() || !setB.emptyOrder())
100
101
         {
             if (A_elt == B_elt)
102
103
                 setAB.pushOrder(A_elt);//collect their common elements
104
105
                 A_elt = setA.popOrder();// go to next elt of setA
106
107
             else {
                 if (A_elt < B_elt) {</pre>
108
109
                     setAB.pushOrder(A_elt);
                     A_elt = setA.popOrder();//look at the next element of setA
110
111
112
                 }
                 else {
113
114
115
                     B_elt = setB.popOrder();//look at the next element of setB
116
                     setAB.pushOrder(B_elt);
                 }
117
             }
118
119
         }
120
121
         //display their union set, setAB
         cout << "A union B = "; setAB.displayOrder();</pre>
122
123
         return 0;
124 }
125
126
127
```