Name: 周嘉禾

Student ID: D1166506

In this program, I used recursively node to initialize the node and defined the node pointer as “List”. Next, I filled the function with the code:

1. Initial
   Let List *L=NULL.
2. getSize
   Traverse the elements of the List L and return the size.
3. getElem
   Traverse to the assigned position and return the element at that position.
   Return -1 while there is any unwanted situation.
4. setElem
   Traverse to the assigned position, change the element at that position, and return 1 as setting the element successfully.
   Return -1 while there is any unwanted situation.
5. search
   If find the element required, return the position of the element.
   Otherwise, return -1 as cannot find the element.
6. insertElem
   If the element wanted to insert the list had existed, return -1 as inserting failed.
   Otherwise, create a node, change *L or next to the node, and return the position of the node inserting.
7. deleteElem
   Traverse to find the position where the node locating, change the next of previous node, and return the element at that position.
   Return -1 while there is any unwanted situation.
8. printList
   Traverse the list and print all the element inside the list.
   Create a newline every 20 elements.
9. append
   Since there aren’t any repeated elements inside the L1, we can plug all the elements into L3 at first. Then, use search function to skip the elements repeated in L1 and plug other elements into L3.

At the end, return the List L3.

10. join

    Traverse L1 and use search function to find if there is any element repeated. If the element can be found in both L1 and L2, plug the element into the List L4.

    Then, return the List L4.

11. sort

    Traverse the List and rearrange the elements of the List. If the current one is greater than the next one, swap them.

    After the rearrangement, the List can be guaranteed that it is in ascending order.

At the end, in the main function, I just printed some sentences required and called the functions needed as the instruction.