WAP to Implement Single Link List with following operations: Sort the linked list, Reverse the linked list, Concatenation of two linked lists.

Code:

```
#include <stdio.h>
#include <stdlib.h>

struct Node {
   int val;
   struct Node* next;
};

void sortList(struct Node** node);
void create(struct Node** node);
void display(struct Node** node);
void insert(struct Node** node, int value);
void reverse(struct Node** node);
void concat(struct Node** node1, struct Node** node2);

int main() {
   struct Node* head1 = NULL;
   struct Node* head2 = NULL;
   printf("Create LL 1 : \n");
   create(&head1);
   printf("Create LL 2 : \n");
   create(&head2);

   printf("Concatination of two lists is : \n");
   concat(&head1, &head2);
   display(&head1);

   printf("Sorting of this list : \n");
   sortList(&head1);
   display(&head1);

   printf("Reversing of this list : \n");
   reverse(&head1);
}

void create(struct Node** node) {
   int ch, val;
   while (1) {
      printf("1. Insert\n2. Exit\n");
      scanf("%d", &ch);
```

```
      switch (ch) {
         case 1:
            printf("Enter the value : ");
            scanf("%d", &val);
            insert(node, val);
            break;
         case 2:
            return;
         default:
            printf("Invalid choice\n");
      }
   }
}

void insert(struct Node** node, int value) {
   struct Node* new_node = (struct Node*)malloc(sizeof(struct Node));
   new_node->val = value;
   new_node->next = *node;
   *node = new_node;
}

void sortList(struct Node** node) {
   struct Node *temp, *i;
   for (temp = *node; temp != NULL; temp = temp->next) {
      for (i = temp->next; i != NULL; i = i->next) {
         if (i->val < temp->val) {
            int tem = i->val;
            i->val = temp->val;
            temp->val = tem;
         }
      }
   }
}
void display(struct Node** node) {
   struct Node* temp = *node;
   while (temp != NULL) {
      printf("%d->", temp->val);
      temp = temp->next;
   }
   printf("NULL\n");
}
void reverse(struct Node* *node) {
   struct Node* temp = *node;
   struct Node* curr = temp;
```

```
    struct Node* prev = NULL;
    struct Node* nextOne = NULL;

    while(curr != NULL) {
        nextOne = curr->next;
        curr->next = prev;
        prev = curr;
        curr = nextOne;
    }
    display(&prev);
}
void concat(struct Node* *node1, struct Node* *node2) {
    struct Node* temp1 = *node1;
    struct Node* temp2 = *node2;

    struct Node* dummy = temp1;
    while(dummy->next != NULL) dummy = dummy->next;

    dummy->next = temp2;
}
```

Output:

```
Create LL 1 :
1. Insert
2. Exit
1
Enter the value : 2
1. Insert
2. Exit
1
Enter the value : 3
1. Insert
2. Exit
1
Enter the value : 4
1. Insert
2. Exit
1
Enter the value : 5
1. Insert
2. Exit
1
Enter the value : 6
1. Insert
2. Exit
2
Create LL 2 :
1. Insert
2. Exit
1
```

```
Enter the value : 3
1. Insert
2. Exit
1
Enter the value : 5
1. Insert
2. Exit
1
Enter the value : 4
1. Insert
2. Exit
1
Enter the value : 2
1. Insert
2. Exit
2
Concatination of two lists is :
6->5->4->3->2->2->4->5->3->NULL
Sorting of this list :
2->2->3->3->4->4->5->5->6->NULL
Reversing of this list :
6->5->5->4->4->3->3->2->2->NULL
```