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

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

struct Node* head1 = NULL;
struct Node* head2 = NULL;

void insertEnd(struct Node** head, int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node
        ));
    newNode->data = value;
    newNode->next = NULL;

    if (*head == NULL) {
        *head = newNode;
        return;
    }

    struct Node* temp = *head;
    while (temp->next != NULL)
        temp = temp->next;
    temp->next = newNode;
}
```

```
1. Insert
2. Display
3. Sort
4. Reverse
5. Concatenate Two Lists
6. Exit
1
Insert into list (1/2):
1
Enter value:
12

1. Insert
2. Display
3. Sort
4. Reverse
5. Concatenate Two Lists
6. Exit
1
Insert into list (1/2):
2
Enter value:
13

1. Insert
2. Display
```

```c
void display(struct Node* head) {
    struct Node* temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

void sortList(struct Node* head) {
    if (head == NULL) return;

    struct Node* i, *j;
    int temp;

    for (i = head; i->next != NULL; i = i->next) {
        for (j = i->next; j != NULL; j = j->next) {
            if (i->data > j->data) {
                temp = i->data;
                i->data = j->data;
                j->data = temp;
            }
        }
    }
}
```

```
3. Sort
4. Reverse
5. Concatenate Two Lists
6. Exit
1
Insert into list (1/2):
2
Enter value:
13

1. Insert
2. Display
3. Sort
4. Reverse
5. Concatenate Two Lists
6. Exit
5
Concatenating List 2 to List 1...

1. Insert
2. Display
3. Sort
4. Reverse
5. Concatenate Two Lists
6. Exit
2
Display which list (1/2):
```

```c
struct Node* reverseList(struct Node* head) {
    struct Node* prev = NULL;
    struct Node* curr = head;
    struct Node* next = NULL;

    while (curr != NULL) {
        next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
    return prev;
}

struct Node* concatenate(struct Node* head1, struct Node* head2) {
    if (head1 == NULL) return head2;
    struct Node* temp = head1;
    while (temp->next != NULL)
        temp = temp->next;
    temp->next = head2;
    return head1;
}
```

```c
int main() {
    int choice, value, listChoice;

    while (1) {
        printf("\n1. Insert\n2. Display\n3. Sort\n4. Reverse\n5.
            Concatenate Two Lists\n6. Exit\n");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Insert into list (1/2):\n");
                scanf("%d", &listChoice);
                printf("Enter value:\n");
                scanf("%d", &value);
                if (listChoice == 1)
                    insertEnd(&head1, value);
                else
                    insertEnd(&head2, value);
                break;

            case 2:
                printf("Display which list (1/2):\n");
                scanf("%d", &listChoice);
                if (listChoice == 1)
                    display(head1);
                else
```

```
        case 3:
            printf("Sort which list (1/2):\n");
            scanf("%d", &listChoice);
            if (listChoice == 1)
                sortList(head1);
            else
                sortList(head2);
            break;

        case 4:
            printf("Reverse which list (1/2):\n");
            scanf("%d", &listChoice);
            if (listChoice == 1)
                head1 = reverseList(head1);
            else
                head2 = reverseList(head2);
            break;

        case 5:
            printf("Concatenating List 2 to List 1...\n");
            head1 = concatenate(head1, head2);
            head2 = NULL;
            break;
```

```
            case 6:
                return 0;
        }
    }
}
```