```
#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;
}

void display(struct Node* head) {
   if (head == NULL) {
      printf("List is empty!\n");
      return;
   }

   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 || (*head)->next == NULL) return;

   struct Node *i, *j;
   int temp;

   for (i = *head; i != 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;
          }
        }
    }
}

void reverseList(struct Node** head) {
    struct Node *prev = NULL, *current = *head, *next = NULL;

    while (current != NULL) {
        next = current->next;
        current->next = prev;
        prev = current;
        current = next;
    }
    *head = prev;
}

void concatenateLists() {
    if (head1 == NULL) {
        head1 = head2;
    } else if (head2 != NULL) {
        struct Node* temp = head1;
        while (temp->next != NULL)
            temp = temp->next;
        temp->next = head2;
    }
    head2 = NULL;
    printf("Concatenating List 2 to List 1...\n");
}

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

    while (1) {
        printf("\n1. Insert\n");
        printf("2. Display\n");
        printf("3. Sort\n");
        printf("4. Reverse\n");
        printf("5. Concatenate Two Lists\n");
        printf("6. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Insert into list (1/2): ");
                scanf("%d", &listChoice);
                printf("Enter value: ");
                scanf("%d", &value);
```

```
        if (listChoice == 1) {
            insertEnd(&head1, value);
        } else if (listChoice == 2) {
            insertEnd(&head2, value);
        } else {
            printf("Invalid list choice! Choose 1 or 2\n");
        }
        break;

    case 2:
        printf("Display which list (1/2): ");
        scanf("%d", &listChoice);

        if (listChoice == 1) {
            printf("List 1: ");
            display(head1);
        } else if (listChoice == 2) {
            printf("List 2: ");
            display(head2);
        } else {
            printf("Invalid list choice! Choose 1 or 2\n");
        }
        break;

    case 3:
        printf("Sort which list (1/2): ");
        scanf("%d", &listChoice);

        if (listChoice == 1) {
            sortList(&head1);
            printf("List 1 sorted.\n");
        } else if (listChoice == 2) {
            sortList(&head2);
            printf("List 2 sorted.\n");
        } else {
            printf("Invalid list choice! Choose 1 or 2\n");
        }
        break;

    case 4:
        printf("Reverse which list (1/2): ");
        scanf("%d", &listChoice);

        if (listChoice == 1) {
            reverseList(&head1);
            printf("List 1 reversed.\n");
        } else if (listChoice == 2) {
            reverseList(&head2);
            printf("List 2 reversed.\n");
        } else {
            printf("Invalid list choice! Choose 1 or 2\n");
```

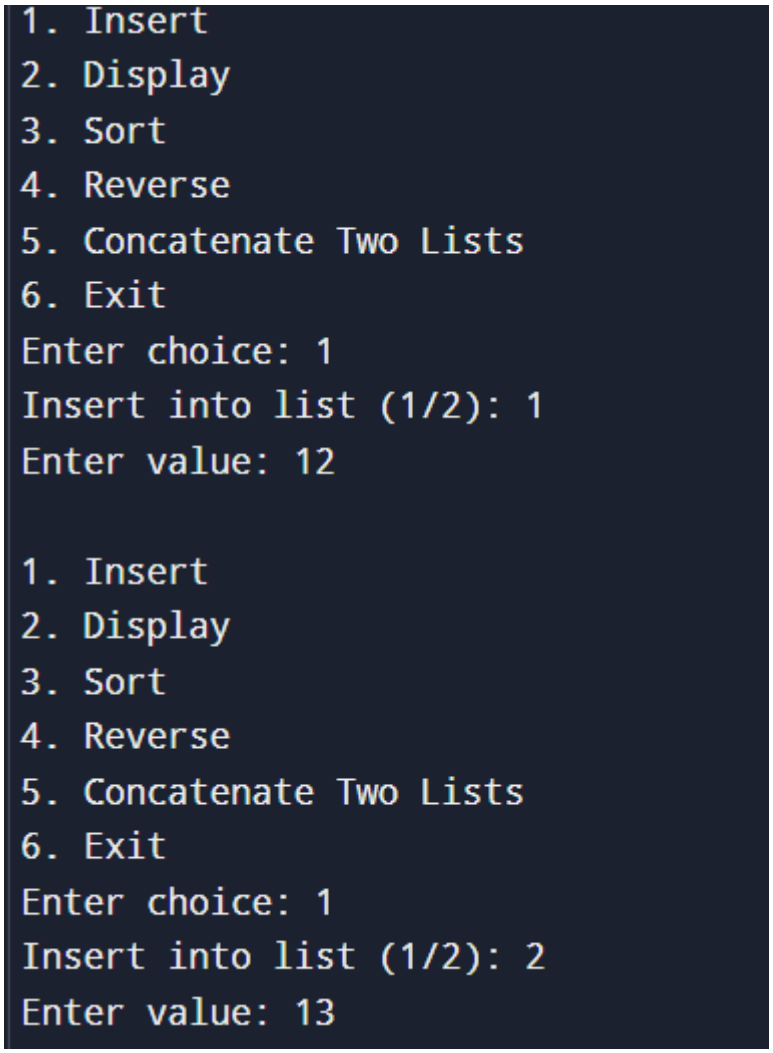
```
            }
            break;

        case 5:
            concatenateLists();
            break;

        case 6:
            exit(0);

        default:
            printf("Invalid choice!\n");
      }
    }

    return 0;
}
```



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

1. Insert
2. Display
3. Sort
4. Reverse
5. Concatenate Two Lists
6. Exit
Enter choice: 2
Display which list (1/2): 1
List 1: 12 -> 13 -> NULL
```