7) WAP implement single link list with following operations.

a) Sort the linked list
b) Reverse the linked list
c) Concatenation of two linked lists.

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

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

void create(struct node **hptr){

    struct node *newnode, *temp;
    int item;

    newnode = (struct node *) malloc(sizeof
                (struct node));
    printf("Enter the data");
    scanf("%d", &item);

    newnode->data = item;
    newnode->next = NULL;

    if (*hptr == NULL){
        *hptr = newnode;
    }
    else {
        temp = *hptr;
```

```
        while (temp->next != NULL) {
            temp = temp->next;
        }
        temp->next = newnode;
    }
}

void display (struct node *hptr) {

    struct node *temp;
    temp = hptr;

    if (hptr == NULL) {
        printf ("The list is empty");
    }
    else {
        while (temp != NULL) {
            printf ("%d", temp->data);
            printf ("->");
            temp = temp->next;
        }
    }
}

void reverse (struct node **hptr) {

    struct node *prev = NULL, *current
    = *hptr, *next = NULL;

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

```
    *hptr = prev;
}

void concat (struct node *temp1,
 struct node *temp2) {

    while (temp1->next != NULL) {
        temp1 = temp1->next;
    }
    temp1->next = temp2;

}

void sort (struct node *hptr) {

    struct node *current = NULL, *next = NULL;
    current = hptr;
    int temp;

    while(current->next != NULL){
        next = current->next;

        while(next != NULL) {

            if ((current->data) > (next->data)){
            temp = current->data;
            current->data = next->data;
            next->data = temp;
            }

            next = next->next;
        }

        current = current->next;
    }
}
```