```
//Created By Ritwik Chandra Pandey on 03/03/21
//183215
//Max,min,median

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

/* Structure for a node */
struct node
{
    float data;
    struct node* next;
};

/* Function to insert a node */

void addNodes(struct node **first, float x) {
    struct node *temp = (struct node*) malloc(sizeof(struct node));
    struct node *lastNode = *first;

    temp -> data = x;
    if ((*first) == NULL) {
        (*first) = temp;
    } else {
        while (lastNode -> next != NULL) {
            lastNode = lastNode -> next;
        }
        lastNode -> next = temp;
    }


}


/* Function to print nodes */
void display_list(struct node *node)
{
    while (node!=NULL)
    {
        printf("%f ", node->data);
```

```
        node = node -> next;
    }
}
struct node* sort(struct node *first) {
    struct node *t1,*t2;
    float x;
    for(t1=first;t1->next!=NULL;t1=t1->next){
        for(t2=t1->next;t2!=NULL;t2=t2->next){
            if(t1->data>t2->data){
                x=t1->data;
                t1->data=t2->data;
                t2->data=x;
            }
        }
    }
    return first;

}


/* Function to remove duplicates from a sorted list */
void printMedian(struct node* head)
{ head = sort(head);
    struct node* slow_ptr = head;
    struct node* fast_ptr = head;
    struct node* pre_of_slow = head;

    if (head != NULL) {
        while (fast_ptr != NULL && fast_ptr->next != NULL) {

            fast_ptr = fast_ptr->next->next;

            // previous of slow_ptr
            pre_of_slow = slow_ptr;
            slow_ptr = slow_ptr->next;
        }

        // if the below condition is true linked list
        // contain odd Node
        // simply return middle element
```

```
        if (fast_ptr != NULL)
            printf("Median is %f\n",slow_ptr->data);

        // else linked list contain even element
        else
            printf("Median is : %f\n",(slow_ptr->data + pre_of_slow->data)/2.0);
    }
}

void largestElement(struct node* head)
{float max;
    if(head!=NULL){
         max = head->data;}

    // Check loop while head not equal to NULL
    while (head != NULL) {

        if (max < head->data)
        { max = head->data;}
        head = head->next;
    }
printf("Maximum is %f\n",max);
}
void smallestElement(struct node* head)
{float min;
    if(head!=NULL){
        min = head->data;}

    // Check loop while head not equal to NULL
    while (head != NULL) {

        if (min > head->data)
        { min = head->data;}
        head = head->next;
    }
    printf("Minimum is %f\n",min);
}
int main()
{
    struct node* head = NULL;
```

```
    float x;

    printf("Enter elements up to -1 : ");
    scanf("%f", &x);
    while (x != -1) {
        addNodes(&head,x);
        scanf("%f", &x);
    }

    display_list(head);
    printf("\n");
    printMedian(head);
    largestElement(head);
    smallestElement(head);


    return 0;
}
```