Mahender Singh

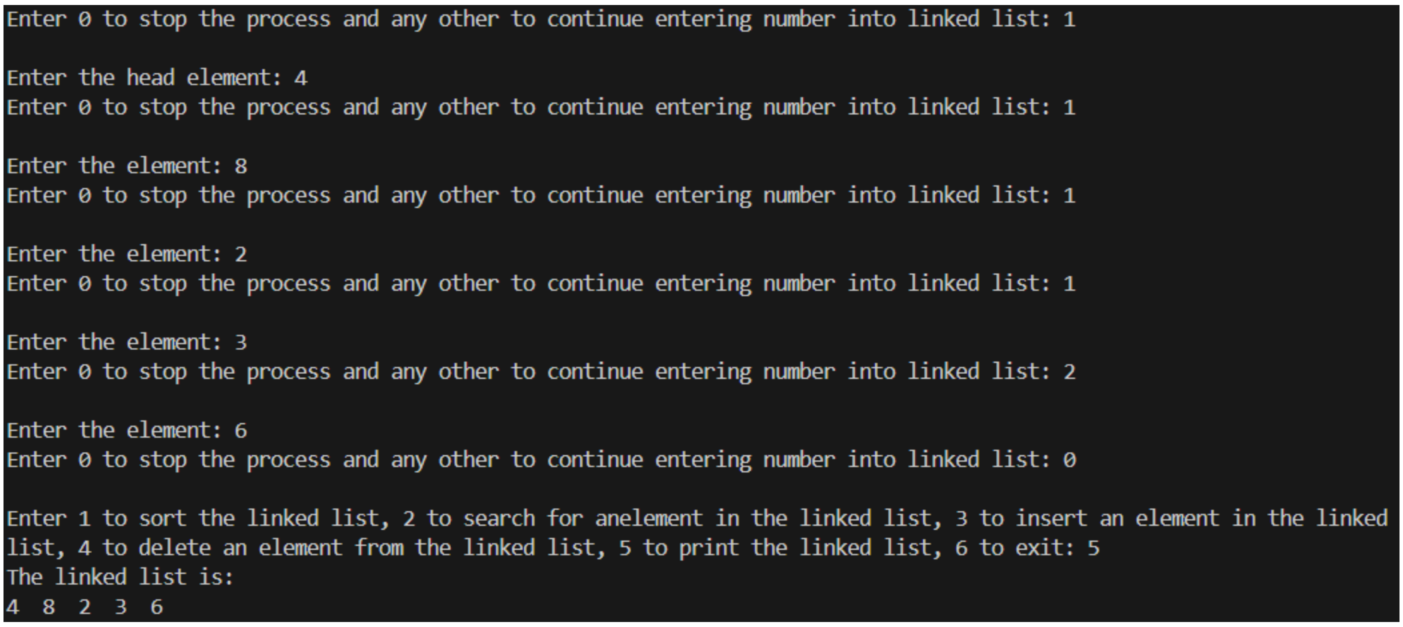
22/11/EC/036

Implementing Kernel Data Structure – Linked List

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

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

void printLinkedList(struct Node *head){
    printf("The linked list is:\n");
    struct Node *p = head;
    do{
        printf("%d  ", p->data);
        p = p->next;
    }while (p!=NULL);
};

void searchLinkedList(struct Node *head, int element){
    struct Node *ptr = head;
    int num = 0;
    while (ptr!=NULL)
    {
        if (ptr->data == element)
        {
            printf("Element found at node at %d",num);
            return;
        }
        ptr= ptr->next;
        num++;
    }
    printf("\nElement was not found in the Linked List");
}

struct Node *insertElement(struct Node *head){
    int userChoice;
    printf("\nEnter 1 to insert element at head, 2 to insert element at an index and 3 to 
insert element at the end ");
    scanf("%d", &userChoice);
    printf("\nEnter the element to be inserted");
    int element;
    scanf("%d", &element);
    if (userChoice==1)
    {
        struct Node *ptr= (struct Node *)malloc(sizeof(struct Node));
        ptr->data= element;
        ptr->next = head;
        head= ptr;
    }
    else if (userChoice==2)
    {
        struct Node *ptr= head;
        int index;
        printf("\nEnter the index you want to insert element at: ");
        scanf("%d", &index);
        int count = 0;

        while (count<index-1)
        {
            ptr=ptr->next;
            count++;
        }

        struct Node *p = (struct Node *)malloc(sizeof(struct Node));
        p -> next = ptr->next;
        ptr->next = p;
        p->data = element;
    }
    else if (userChoice==3)
    {
        struct Node *ptr = (struct Node *)malloc(sizeof(struct Node));
        struct Node *p = head;
        while (p->next!=NULL)
        {
            p = p->next;
        }

        p->next = ptr;
        ptr -> data = element;
        ptr -> next = NULL;
    }
    else{
        printf("\nEnter a valid choice");
    }
    return head;

}

struct Node * deleteElement(struct Node *head){
    printf("\n Enter 1 to delete head node, 2 to delete a node at a given index and 3 to
delete the last node: ");
    int choice;
    scanf("%d", &choice);
    if (choice==1)
    {
        struct Node *temp = head->next;
        free(head);
        head = temp;
    }
    else if (choice==2)
    {
        int i;
        printf("\nEnter the index you want to delete the node of: ");
        scanf("%d", &i);
        int c=0;

        struct Node *temp = head;
        while (c<i-1)
        {
            temp= temp->next;
            c++;
        }

        struct Node *t2 = temp->next;
        temp->next=t2->next;
        free(t2);

    }
    else if (choice==3)
    {
        struct Node *temp = head;
        while (temp->next->next!=NULL)
        {
            temp=temp->next;
        }

        struct Node *temp2 = temp->next;
        temp->next = NULL;
        free(temp2);
    }
    else{
        printf("\nEnter a valid choice");
    }

    return head;

}

void sortLinkedList(struct Node *head){
    struct Node *temp = head;
    int c=0;
    while (temp!=NULL)
    {
        temp=temp->next;
        c++;
    }

    int i=1;
    temp = head;
    while (i<c)
    {
        int j=0;
        temp = head;
        while (j<=c-i-1)
        {
            struct Node *temp2;
            temp2 = temp->next;

            if (temp->data >= temp2->data)
            {
                int x = temp->data;
                temp->data = temp2->data;
                temp2->data= x;
            }
            temp = temp->next;
            j++;

        }
        i++;

    }

}

int main(){
    int stop = 1;
    int i=0;

    struct Node *head;
    struct Node *present;

    while (stop!=0)
    {
        printf("Enter 0 to stop the process and any other to continue entering number into
linked list: ");
        scanf("%d", &stop);

        if (stop==0)
        {
            break;
        }
        else{
            i++;
            if (i==1)
            {
                head = (struct Node *)malloc(sizeof(struct Node));
                int input;
                printf("\nEnter the head element: ");
                scanf("%d", &input);
                head ->data = input;
                head->next = NULL;
                present = head;
            }
            else{
```

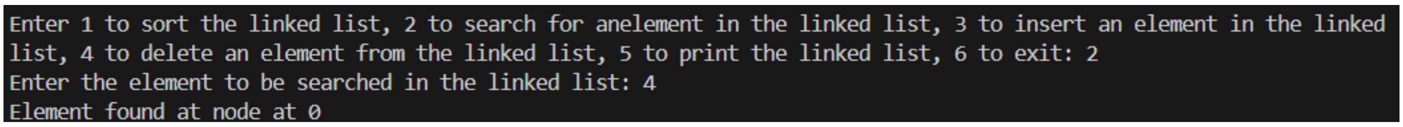
```c
                struct Node * ptr = (struct Node *)malloc(sizeof(struct Node));
                int input;
                printf("\nEnter the element: ");
                scanf("%d", &input);
                ptr->data = input;
                ptr->next= NULL;
                present->next = ptr;
                present = ptr;
            }

        }

    }

    while (1)
    {
        printf("\nEnter 1 to sort the linked list, 2 to search for anelement in the linked
list, 3 to insert an element in the linked list, 4 to delete an element from the linked
list, 5 to print the linked list, 6 to exit: ");
        int choice;
        scanf("%d", &choice);
        if (choice==1)
        {
            sortLinkedList(head);
            printLinkedList(head);
        }
        else if (choice==2)
        {
            printf("Enter the element to be searched in the linked list: ");
            int element;
            scanf("%d", &element);
            searchLinkedList(head, element);
        }
        else if (choice==3)
        {
            head = insertElement(head);
        }
        else if (choice==4)
        {
            head = deleteElement(head);
        }
        else if (choice==5)
        {
            printLinkedList(head);
        }
        else if (choice ==6)
        {
            break;
        }
        else{
            printf("\nEnter a valid choice");
        }

    }
    return 0;

}
```

Creating a Linked List



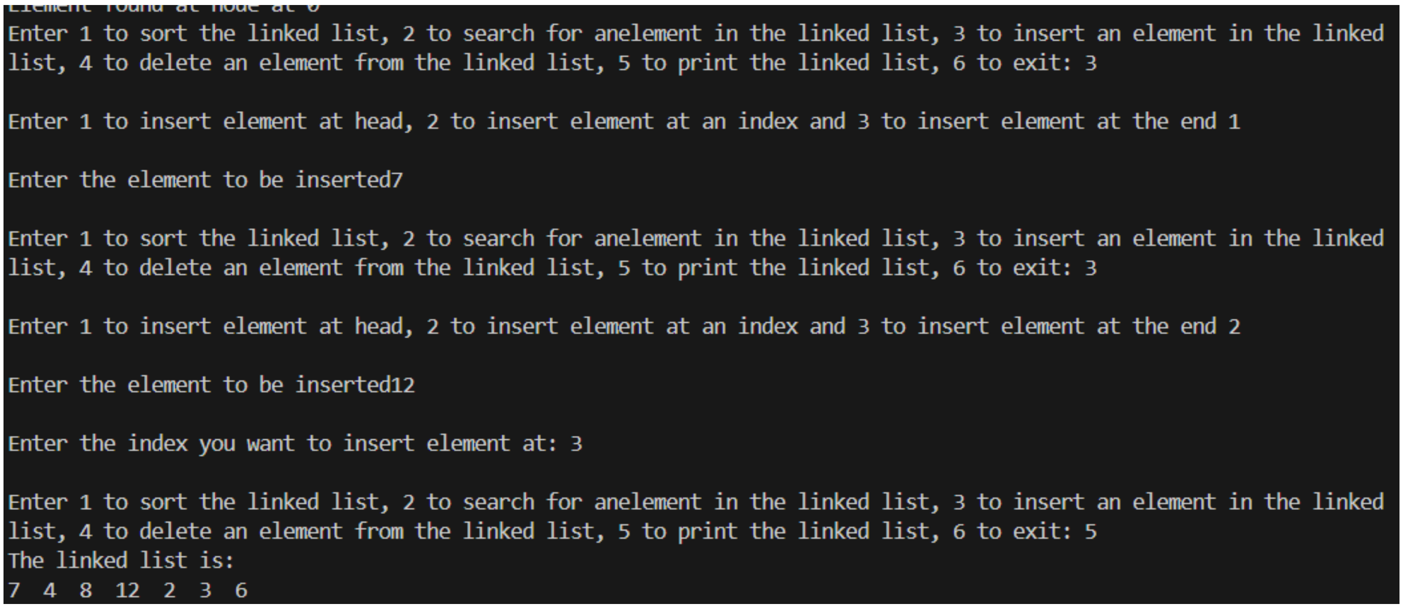
```
Enter 0 to stop the process and any other to continue entering number into linked list: 1

Enter the head element: 4
Enter 0 to stop the process and any other to continue entering number into linked list: 1

Enter the element: 8
Enter 0 to stop the process and any other to continue entering number into linked list: 1

Enter the element: 2
Enter 0 to stop the process and any other to continue entering number into linked list: 1

Enter the element: 3
Enter 0 to stop the process and any other to continue entering number into linked list: 2

Enter the element: 6
Enter 0 to stop the process and any other to continue entering number into linked list: 0

Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 5
The linked list is:
4  8  2  3  6
```

Searching an element



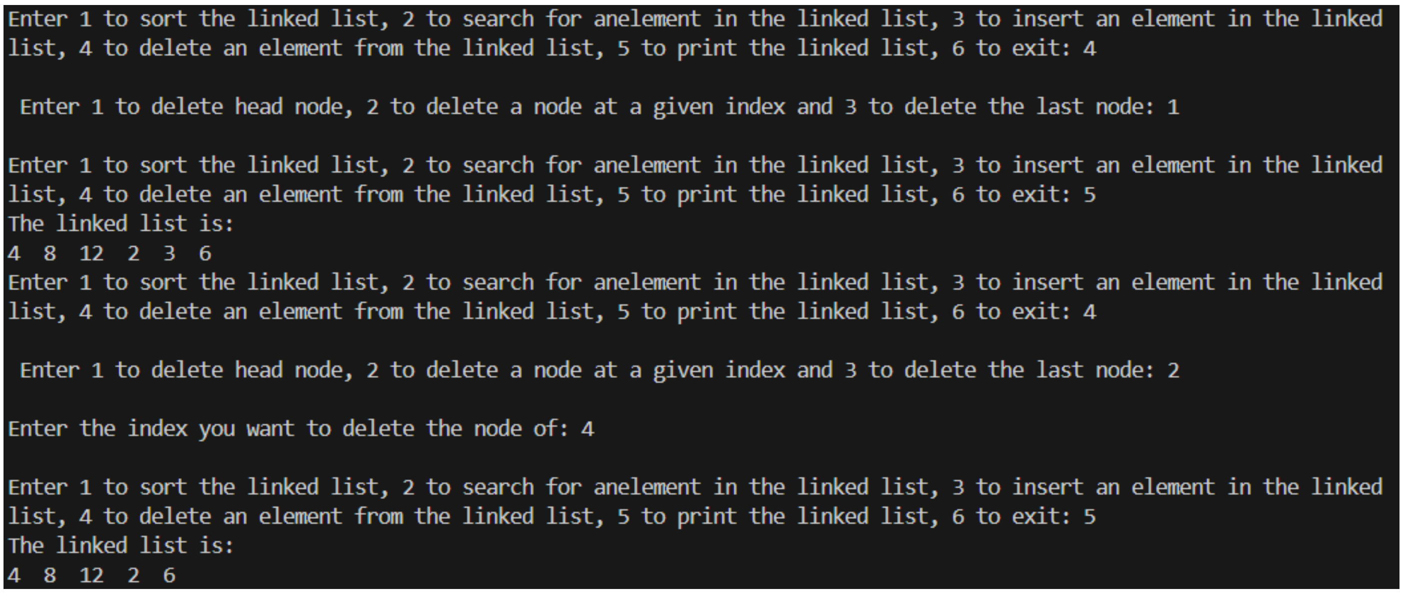
```
Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 2
Enter the element to be searched in the linked list: 4
Element found at node at 0
```

Inserting an element



```
Element found at node at 0
Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 3

Enter 1 to insert element at head, 2 to insert element at an index and 3 to insert element at the end 1

Enter the element to be inserted7

Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 3

Enter 1 to insert element at head, 2 to insert element at an index and 3 to insert element at the end 2

Enter the element to be inserted12

Enter the index you want to insert element at: 3

Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 5
The linked list is:
7  4  8  12  2  3  6
```

Deleting an element



```
Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 4

 Enter 1 to delete head node, 2 to delete a node at a given index and 3 to delete the last node: 1

Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 5
The linked list is:
4  8  12  2  3  6
Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 4

 Enter 1 to delete head node, 2 to delete a node at a given index and 3 to delete the last node: 2

Enter the index you want to delete the node of: 4

Enter 1 to sort the linked list, 2 to search for anelement in the linked list, 3 to insert an element in the linked
list, 4 to delete an element from the linked list, 5 to print the linked list, 6 to exit: 5
The linked list is:
4  8  12  2  6
```