Date 05/02/2024
Page

lab-7

1) Double linked list b) insert node to left of the node c) delete node node bassed on value.

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

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

struct Node * create( int n){
      struct Node* head;
      struct Node * tail;
      for( int i=0; i<n; i++){
            int data;
            printf("Enter data for node %d:",
                                    i+1);
            scanf("%d", &data);
            struct Node * newNode= (struct Node*)
                  malloc( sizeof (struct Node));
            newNode->data = data;
            newNode-> prev= tail;
            newNode->next= NULL;
            if( tail !=NULL){
                  tail->next=newNode;
            }
            else{
                  head=newNode;
            }
            tail = newNode;
      }
      return head;
}
```

```
void display(struct Node* head){
      struct Node* current=head;
      printf("Double Linked List:\t");
      while(current!=NULL){
            printf("%d\t", current->data);
            current=current->next;
      }
      printf("\n");
}

void insert(struct Node** head, int targetData,
                 int newData);
      struct Node* current = *head;
      while(current!=NULL){
            if (current->data==targetData){
                  struct node* newNode = (struct
        Node*)malloc(sizeof(struct Node));
                  newNode->data=newData;
                  newNode->prev= current->prev;
                  newNode->next=current;
                  if(current->prev!=NULL){
                        current->prev->next=
                                          newNode;
                  }
                  else{
                        *head=newNode;
                  }
                  current->prev=newNode;
                  printf("Node with data %d ins
-erted to the left of node with data %d\n",
                              newData,targetData);
                  return;
            }
            current=current->next;
      }
```

```
        printf("Node with data %d not found\n"
                                  , targetData);

void delete( struct Node **head, int key){
    struct Node *current = *head;
    while ( current!=NULL){
        if (current->data ==key ){
            if (current->prev!=NULL){
                current->prev->next= current
                                       ->next;
            }
            else {
                *head = current->next;
            }
            if ( current->next!=NULL ){
                current->next->prev =current
                                      ->prev;
            }
            free(current);
            printf("Node with data %d
                         deleted\n", key );
            return;
      }
      current=current->next;
    }
    printf("Node with data %d not found\n"
                             , key);
}
```

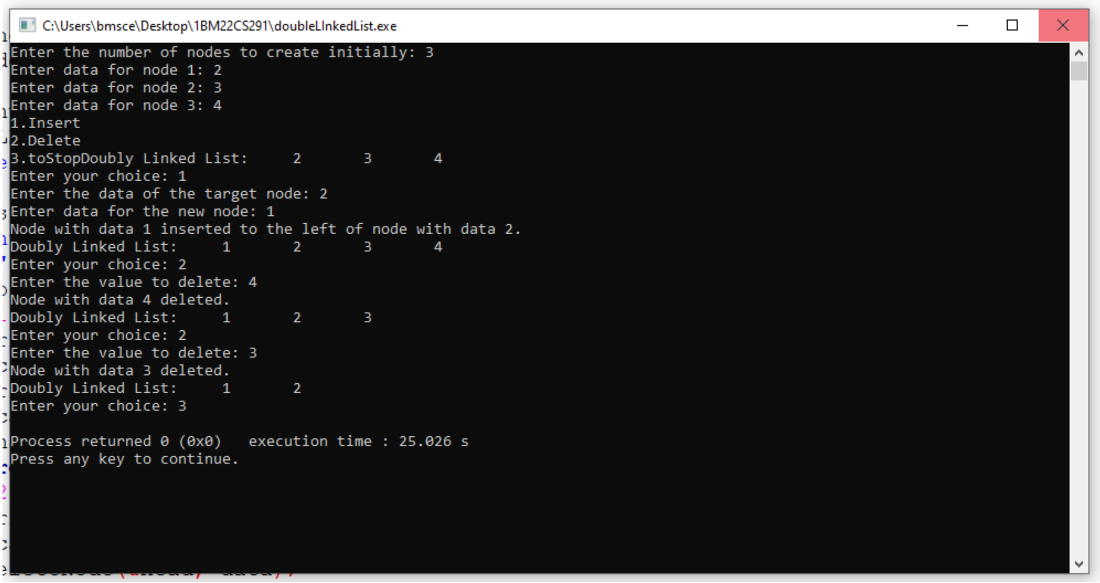
```
int main(){
        struct Node *head=NULL;
        int op, num, data, target Data;
        printf("Enter the number of nodes to
                    create \n");
        scanf("%d", &num);
        head = create( (num);
        printf(" Enter 1. Insert \n 2. Delete\n
                    3. Exit \n");
        while(1){
                display(head);
                printf("Enter operation\n");
                scanf("%d",&op);
                switch (op){
                    case 1:
                        printf("Enter data of the
        target Node \n");
                        scanf("%d", & targetData);
                        printf("Enter data for
                                newNode\n");
                        scanf("%d",&data);
                        insert(&head, target Data,
                                data);
                        break;
                    case 2:
                        printf("Enter the value
                            to delete:\n");
                        scanf("%d", &data);
                        delete( &head, data);
                        break;
                    case 3:
                        exit(0);
                    default:
                        printf("Invalid choice\n");
                }
```

```
    }
    return 0;
}
```

O/P

```
Enter the number of nodes to create
initially: 3
Enter data for node1: 1
Enter data for node2: 2
Enter data for node 3: 3
1. Insert
2. Delete
3. tostop
Doubly Linked list:   1   2   3
Enter your choice: 2
Enter the value to delete: 3
Node with data 3 deleted.
Double linked list:   1   2
Enter your choice:1
Enter data for the target node: 1
Enter data for new node: 0
Node with data 0 inserted to the
left of node with data 1.
Doubly linked list:   0   1   2
Enter choice: 3
```

Output:



```
Enter the number of nodes to create initially: 3
Enter data for node 1: 2
Enter data for node 2: 3
Enter data for node 3: 4
1.Insert
2.Delete
3.toStopDoubly Linked List:     2       3       4
Enter your choice: 1
Enter the data of the target node: 2
Enter data for the new node: 1
Node with data 1 inserted to the left of node with data 2.
Doubly Linked List:     1       2       3       4
Enter your choice: 2
Enter the value to delete: 4
Node with data 4 deleted.
Doubly Linked List:     1       2       3
Enter your choice: 2
Enter the value to delete: 3
Node with data 3 deleted.
Doubly Linked List:     1       2
Enter your choice: 3

Process returned 0 (0x0)   execution time : 25.026 s
Press any key to continue.
```