SUSHMITHA Y.V
1BM19CS165

[illegible]

Add a node to left of a node, delete a node and display a doubly linked list.

```
typedef struct Node {
        int value;
        struct Node * next;
        struct Node * prev;
    }
     node;
    node * head = NULL;
    void add = beg (int value) // add at beginning

{
   node * ptr = (node *) malloc (size of (node));
          ptr -> value = value;
          ptr -> prev = NULL;
          ptr -> next = head;
        if (head != NULL)
          head -> prev = ptr;
           head = ptr;
       }
```

```
void add_key (int value, int key) // add behind
                                          key

{ node * temp = head;
    while (tmp != NULL) {
        if ( tmp -> value = = key)
            break;
        tmp = tmp -> next;
    }
    if ( tmp = = head)
    {
        add_beg (value);
        return;
    }
    node * ptr = (node *) malloc( size of (node));
        ptr -> value = value;
        ptr -> prev = tmp -> prev;
            ptr -> next = tmp;
        (tmp -> prev) -> next = temp;
            tmp -> prev = ptr;
}
```

```
void del_Key (int Key) {
    if (head == NULL) {
      printf (" list is empty ");
          return;
    }

  node * tmp = head;
   while (tmp! = NULL) {
        if (tmp -> value = = key)
              break;
         tmp = tmp -> next;
   }
     if (tmp = = head)

    {
       if (head -> next = = NULL)
        {
           free (head);
            head = NULL;
             return;
        }
head = head → next;
 free (head → prev);
  head → prev = null;
   return;
}
 if (tmp → next = NULL)
 {
   tmp → prev → next = NULL;
       free(tmp);
        return;
 }
tmp → next → prev = tmp → prev;
tmp → prev → next = tmp → next;
      free (tmp);
}
  void display ()
{
     if (head == NULL) {
          printf ("list is empty");
               return;
     }
```

```
node * tmp = head;
 printf ("list contains:");
  while (tmp ! =NULL) {
   printf (" %d", tmp -> value);
    tmp = tmp -> next;
  }
}
```