1.> WAP to impliment doubly linked list with primitive operations.

a) Create a double linked list.

```
→ struct node {
        int data;
        struct node * prev;
       struct node * next;
   };
   struct node * head, * tail;
   head = 0;
   void create () {
        struct node * newnode;
newnode = (struct node *) malloc (size of (struct node));
   printf ("Enter data : ");
   Scanf ("%d", & newnode -> data);
      newnode -> next = 0;
      new node -> prev = 0;
  if ( head == 0) {
        head = tail = new node;
    }
 else {
      newnode -> prev = tail;
      tail -> next = newnode;
         tail = newnode;
    }
 }
```

o/p : Enter data: 5

2> Insert a newnode to the ~~left~~ left of a node.

```
→ void newnode (){
struct node * newnode , * ptr ; ptr = head ;
newnode = (struct node *) malloc (size of (struct node));
    printf ("Enter data : ");
    scanf ("%d", &newnode -> data);
    int count = 0 , pos ;
    printf ("Enter position : ");
    Scanf ("%d", &pos)
    while ((count < pos) {
            ptr = ptr -> next ;
            count ++ ;
        }
        newnode -> prev = ptr -> prev ;
        newnode -> next = ptr
        ptr -> prev -> next = newnode
            ptr -> prev = newnode
        }
```

3> Delete the node based on a specific value.

```
->  void  delete () {
      struct node *ptr ; ptr = head ;
      int key , val , flag = 0 ;
      printf ("Enter value to be deleted");
      scanf ("%d", &key );
    while (ptr != NULL) {
      if (ptr->data == key ) {
        ptr -> prev->next = ptr -> next ;
        ptr -> next -> prev = ptr -> prev;
free (ptr);
  break;}
else {
     ptr = ptr->next ;
   }
}
if (flag == 1) {
        printf ("Element deleted");
   }
 else {
        printf ("Element not found");
   }
 }
```

OUTPUT)

```
b) Enter data : 5
   Enter position 2
   Before : 1 9 2
   After = 1 5 9 2
```

[illegible]
5/2/24

```
c) Enter value to be deleted 10.
   Before    1 2 10 5
   after     1 2  5
```