Lab. 9.

Singly Linked List.

```
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#include <process.h>.
```

1. Create a structure node.
   struct node.
2. Getting value of the node.
   NODE getnode().
   {
   NODE x.
   x = (NODE) malloc(sizeof(struct node
   returning NODE. x.
3. Deleteting a node.
   function - void freenode(NODE x.)
   free x.
4. Inserting a node at the front end
   create a node - temp. using
   function getnode().
   Assign data(item) to temp -> info.
   temp -> link = NULL.
   linking this node to front of list.
   temp -> link = first.
   first = temp.
   return the first node.

5. Inserting the node at the end (rear) of the linked list.

function NODE insert_rear(NODE first, int item)

create 2 nodes - temp and cur.
get node temp → temp = getnode().
assign data(item) to temp
temp → info = item
temp = NULL.
cur will be the first node of the list.
looping (transversing through the linked list) while(cur → link != NULL.
cur = cur → link end of while loop.
cur → link = temp.
return the first node.

6. Inserting a node in the middle at a given position

NODE insert_pos(int item, int pos, NODE first)
create 2 nodes temp, prev and cur
get node temp from the function getnode()
assign data (item) to temp
temp → info = item
temp → link = NULL
check for condition if (first = NULL.
this is invalid.
assign count = 1 prev = NULL cur = first.
looping (transversing through linked list to find the position (pos) i.)

while (cur != NULL and count != pos;
prev = cur.
cur = cur → link;
count ++.
end of while loop.
when count is equal to pos.
prev → link = temp } inserting
temp → link = cur. } at pos.
return the first node.

7. Displaying the linked list.
for loop. (from first node until the node → link is NULL.
print the node → info.

8. void main function
Menu - driven statement inside a continuous for loop.
1. Inserting at front 2. Inserting at rear
3. Inserting at a position 4. Display 5. exit
switch (choice)
case 1. Accepts the item from user and calls function insert - front.
case 2 Accepts the item and calls function insert - rear.
case 3. Accepts the item and calls function insert - pos.
case 4. calls function display.
case 5. exits out of the program.