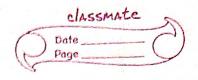
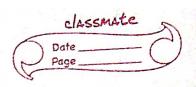
6 5) Singly Linked List Delation Broggran H include < stdio - la > #include < string. h > # include < stalib. h > void create (); void del (chax); void diplay (); void inserthead (); struct node * delete beg (struct node *); struct node * delete end (struct node *); start node f Char nanolio); chan id [10]; int sen; struct node * next; struct node * head = NULL; void rain () & int c; char ele [10]; Printf ("Entra Choice: In 1. Create In 2. Display In 3. Delete a specific IDIA 4. Insert in the beginning In 5. Delete a node from the beginning In 6. Delete a node from the end \n7. Exit \n?"); scarf (" 1/0 d"), &c); switch (c) { Case 1: (reate (); bacak; case 2: display (); break; case 3: printf ("Enter the element id to be deleted in") Start ("1/2 ", ale);



```
del (cle); break;
case 1: inserthead(); break;
case 5: head = delete beg(head); break;
case 6: head : delete_end (head); bacalis
case 7 : exit(o); break;
white (1-);
void create () &
struct node * noconode, * temp;
char n[20], id1[10];
ints;
newnode = (struct node *) nallor (size of (struct node));
paintf ("Enter the Name, USN, Senester: \n");
scanf ("0/0 st?") n);
scand (60 70 5", id1);
sconf ("( 2/0 d') , & s);
stropy (newnode -) name, n);
starpy (newnode ) id , id 1);
neunode > sen = S;
if (head = = NOLL) {
neunode > next = NULL ; 3 ( - - ( ) )
head = newnode;
pentf ("Node is created");
Belse {
 tenp = head;
 while (temp -) next! = NULL)
 temp = temp > rext;
 temp -> next = neumode;
 newrode -> next = NULL;
 partf ("NONE created in");
```



```
temp -> next = del -> next ;
if (del = = NULL)
printf ("Element not found \n"); soturn;
void insorthead () of
stand node *newnode;
chas n[20] id1[10];
printf ("Enter the elements: Name, USN and Sen (n'?) )
scanf (" > s > ? , n) ;
Scarf ("705", id1);
Scanf (" 70 d") &s);
newnode = (struct node *) mallor (size of (struct node))
stropy (neurode -) name, n):
strepy (neurode -> id, id1);
Scanf ( 5 20 d 27, & s);
newnode I next = head;
head : newmode;
struct node *delete_beg (struct node *head) {
struct node * pta;
pts: head;
head = head - to next;
free (ptx);
setion head;
struct node * delete end (struct node *head) {
Steuct node *pts, *prepts;
pte = heads
```

classmate while (ptx > next ! = NOW) & propta = pta;
pta = pta > next;
i # Conc propts -> next = NULL; # in free (ta); Return head;