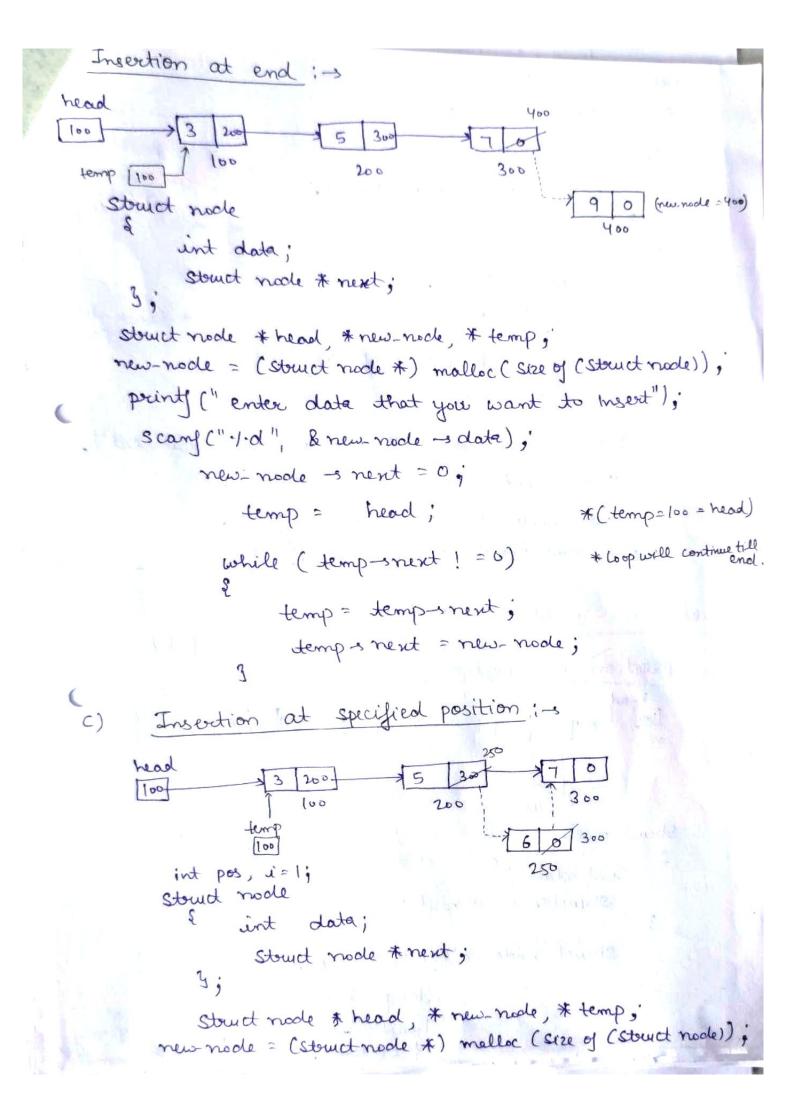
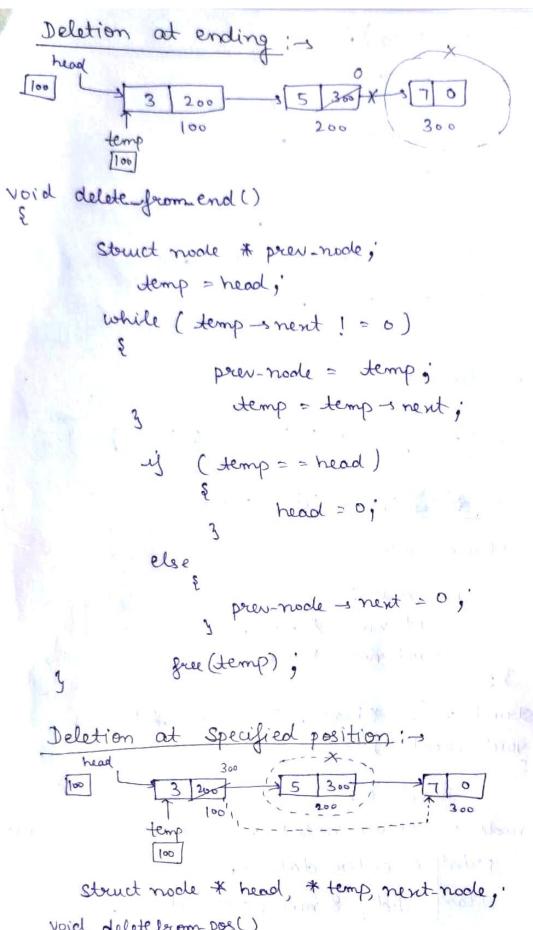


```
temp = head;
11 For display
                while (temp!=0)
 the data
                    pocinty (".1.d", temp -s data);
                    temp = temp = nent;
                getch();
   Insertion in single linked list:
     Insection
                  at beginning
                   > New node insection at begining
 Note: - * (first write the code for creation (as defined previous)
       Struct nocle
             int data;
              Struct node * nent;
        3;
          Struct node * head, * new. node;
        new-node = (struct node * ) malloc ( size of (struct node));
           prints (" enter data that you want to insert");
             Scanf (".1.d", & new-node -solate),
                  new-nocle -s next = head; ] insertion at beginning
  of head = 100
                       head = new-mode;
      Mote! - # Also write code for display. (explained previous)?
```



```
prints (" enter position");
  scang (".1.d", & pos );
   y (pos > count)
                                             * If position is
                                               less than 0 &
        points (" Invalid position");
                                                  greater than
                                                      length.
    else
            temp = head;
             while (+ < pos)
                                               * Reach till
                                                       position
                   temp = temp-s next,
                    i++
        prints (" enter data that you want to insert"),
         Scan ("1.d", & new node -s data),
         new-mode -s next = temp-s next;
       3 temp-next = new-node;
Deletion in Single Linked list;
    Deletion at beginning: -
                  200
              temp [100]
    Struct node
          int date;
          Struct mode * next;
          Struct node * head, * temp,
                           temp = head;
                                                 *deletion at
                            head = head -s next;
                                                      beginning
                            free (temp);
```

4



void deletefrom pos()

int pos, i=1; temp = head; printy (" enter position");

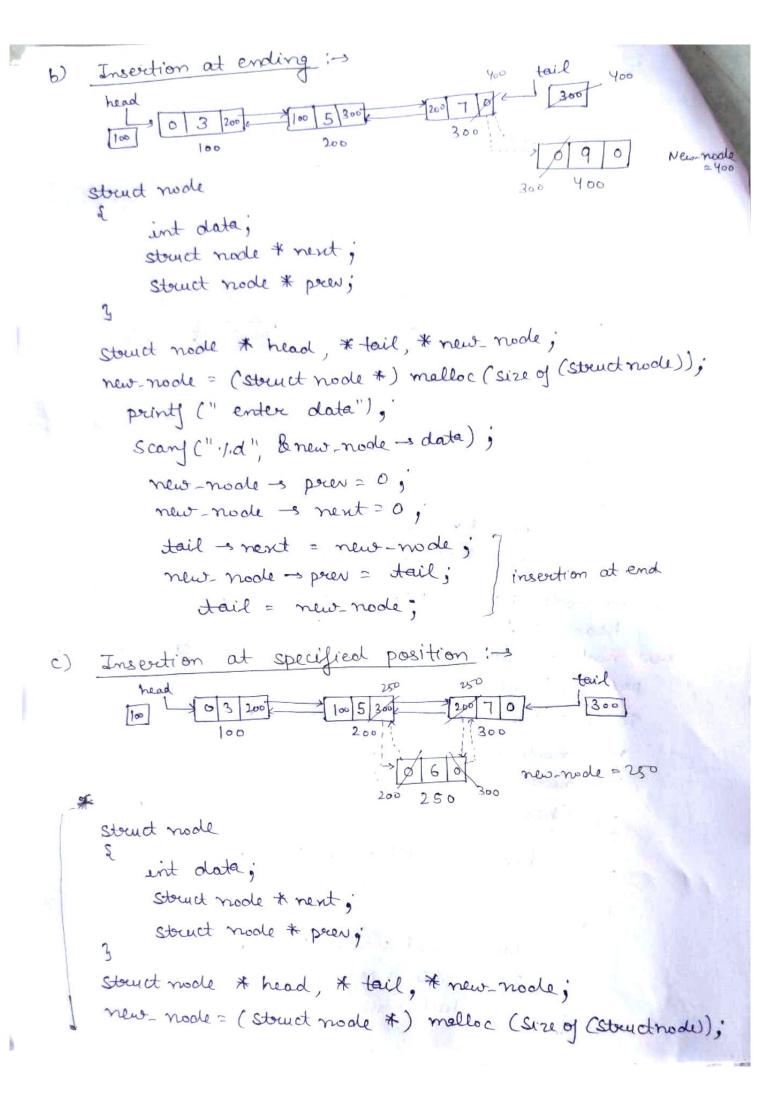
()

```
Scany (". 1.d", & pos);
      while (i < pres-1)
           demp = demp-snext;
              i+t;
       next-node = temp-snext;
        temp-s next = next-node - next;
              free (next-node),
  3
Double Linked list :-
  Creation of double linked list :-
           Struct node
               int data;
                 Struct node * next;
                 Stouct made + prev;
           3;
          Struct node * head, * new-node, * temp;
          void create ()
                 head = 0;
      new-node = (stouct node *) malloc (size of (stouct node));
              prints (" enter data");
              Scan (". I.d", & new-node -s data):
               new node -s pour = 0;
               new-node - next = 0;
            y ( head = = 0)
                    head = temp = new-node;
```

```
temp-snext = new-node;
              new-node - pour = temp;
                 temp = new-node;
                                          * For display, same code as single linked list
Insertion in double linked list :3
  Insection at beginning:
                 3 200 100 5 300 6
                                200
         > 0 1 0 to new-mode
                            * Cuse creation code as explained before
   Struct node
         int data
          Stouct node * next;
          Stouct node * pren;
    Stouct node * head, * tail, * new-node,
        head = 0;
   new-node = (Struct node *) malloc (Size of (Struct node));
       print (" enter data");
       Scanj (" .1. d", & new-node -s data),
        new-node -s pren = 0;
        new-node -s next = 0;
         y ( head = = 0 )
                head = tail = new-node,
          else
                 tail -snert = new-node,
                                                 insertion
                  new-node - pecer = teil;
                       tail = new- node;
              4
```

else

(* use display code as explained »)



```
print ("enter data");
         scanf (" 1.d", & new-node -s date);
          new-node - pecer = 0 ;
          new-node - nent = 0,
           int pos, i=1;
          prints (" enter position");
           scam (" 1/d", & pos);
          while (i < pos-1)
               temp = temp-snext;
                  i++;
            new-node -s prev = temp;
                                              insection at position
            new-mode -s neut = temp-sneut;
              temp -s nent = new node,
   Deletion in double linked list:
a) Deletion at beginning :->
```

void delete from beg ()

Struct mode * temp;

if (head == 0)

printf (" list is empty");

else

temp = head;

head = head = next; head - prier = 0; free (temp); Deletion at ending: b) Void delfromend() Struct node * temp; if (tail = = 0) print (" list is empty"); else temp = tail; tail - prev - nent = 0; tail = tail - prev; tail - s nent = 0; free (temp), Deletion at specified position: 100 5 300 2 200 7 0 void defferompos () int pos, i=1; Stouct node * temp, * head, temp = head;

```
printy (" enter position");
        Scan (" .1. d", 8 pos),
        while ( 1 < pos)
              temp = temp-s next;
               itt ;
          temp - prier - next - temp- next ;
           temp - n'ent spren = temp - pren;
               free (temp);
Circular Linked list:
  Creation of wicular linked list:
         Stouct node
                int data;
                 Stouct mode * next;
            Stouct node * head, * new node, * temp;
               int choice = 1;
                 head = 0;
               while (choice)
           new-node = ( stocut node*) melloc ( Size of (Stocut node));
                 prints (" enter data");
                 Scan (".1.d", & new nocle - data);
                  new-node -s next = 0;
```

if (nead = = 0) head = temp = new node; else temp-s next = new-node; temp = new-node, temp - s next = head; pocint (" Do you want to continue "); scanf (".1.d", & choice), 11 For display Stouct mode * temp; 11 (if not initialized earlies) y (head == 0) prints (" list is empty"); else temp= head; while (temp-snent! = head) pecint (".1.d", temp-s data), temp = temp-s next; preint ("c).d", temp-s date); Insertion in circular linked list: a) Insection at beginning :-

```
Struct node
         int data;
          Stouct node * nent;
      Stouct node * tail, new_node;
    new node = (Struct node *) malloc (Size of (Strent node));
      print ("enter data");
       Scanf ("./.d", & new-node -s data);
             new-node -s next = 0;
         if (tail = = 0)
                 tail = new_node;
                  tail - next = new-node;
          3
         else
             new-node - snext = tail - snext;
               tail - next = new-node;
  * use display code for displaying item.
b) Insertion at ending:
      Struct mode
               unt data,
                Struct node * next;
           Struct node * new-node, * tail;
        new-noall = ( stouct noale *) malloc ( size of ( stouct noale ));
             points (" enter data");
```

Scan (".1.d", & new_ node -e data), new- mode -s neut = 0; ij (tail = =0) tail = new node; tail-snent = new_node; else new node - s next = tail - s next; tail = next = new-node; Insertion at Specified position: -s tail 400 300 250 void unsertations () Stouct node * new-node, * temp; unt pos, i=1, l; preint (" enter position"); scan (".1.d", & pas),. L= getlength(); uf (pos < 0 11 pos > 2) Prints (" invalid position"); else new node = (Stauct node *) malloc (size of (Struct node)); print (" enter data that you want to insert"); Scanj (". I.d", & new-node -s date); new-node - neut = 0;

```
temp = tail -snent;

while (i < pos-1)

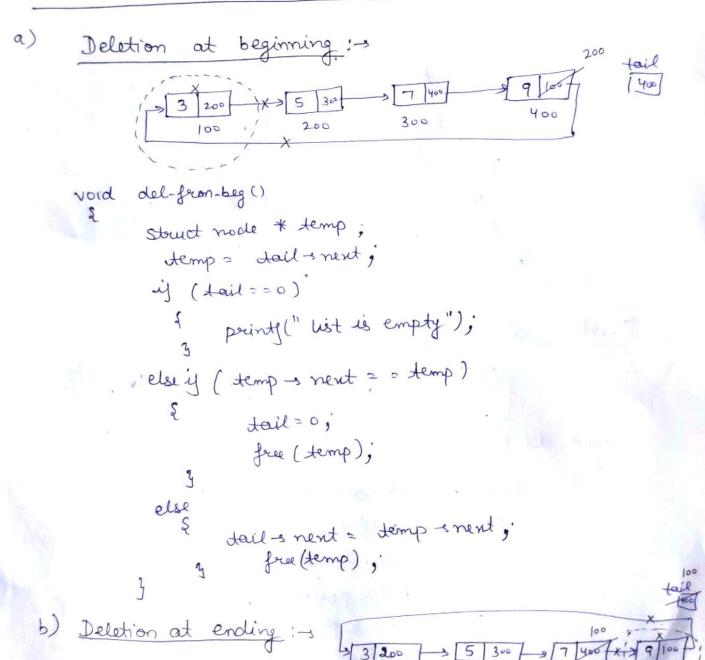
temp = temp-snent;

i++;

new node -sneut = temp-snext;

temp = next = new-node;
```

Deletion in Circular linked list:



```
void del-from-end()
          Struct mode * temp, * prev-mode,
               temp = tail = next;
           ij (tail = =0)
                     perinty (" list is empty");
            else if ( temp-s next = temp)
                     tail = 0;
                      force (temp),
            else
             while (tiemp - s neut! = tail - s neut)
                      prev-node = temp;
                       temp = temp = nent;
                  prev-node -sment = tail-sment;
                       tail = prev- node;
                       free (temp);
        3
C)
    Deletion at Specified position: ->
                3 200 X 5 300 X 7 400 -
      void del-from-pos()
              Struct mode * temp, * next-node;
               int pos, i=1, l;
                  temp = tail-snext;
                print (" enter position"),
                 Scan ["1.d", & pos),
                   l= getlength ();
```

```
if ( pos <1 11 pos>e)
              perinty (" Invalid position");
     else if (pes == 1)
                 del-from-beg();
       3
       else
              while (i < pos-1)
                       temp = temp-snext;
                          i++;
                 nent-node = temp-snent;
                 temp-s next = next-node-snext;
                   free (next_node);
Circular doubly linked list:
                                               tail
                        200 7 400
             100 5 300
                           300
 Insertion :-
 At begin :-
    if (head ==0)
          head = tail = new-node;
           new-node -s prev = tail;
            new-node - s next = head;
     else
             new_node -s next = head;
              nead -s prev = new-node;
               new-node -sprew = tail;
```

```
tail -s next = new- node;
                  head = new- node;
b) At end: -s
     void unsert-at-end()
         if (head = = 0)
                   head = tail = new_ node;
                    new-node -s preen = tail;
                    new-node - next = head;
            3
           else
                  new-node - prier = tail,
                    tail -s nent = new-node;
                     new-rode - next = head;
                      head -> prev = new-node of
                       itail = new-node;
   At Specified position: -s
       void unisert at - pos ()
            Struct node * new-node, to temp
             int pos, i=1, l;
              printy (" enter position");
              scan (".1.d", & pos),
               l = getlength();
              4 (pos <1 11 pos 22)
                     pecinty (" Invalid position"),
               else
                  new-node = Etouctnode *) malloc ( Size of (Studendi));
```

```
prints (" enter data you want to ingert");
            Scanf (".1.d", & new-node = data),
             while (i 2 pos-1)
                      temp = temp-snent;
                         utt;
                  new-node -s prev = temp,
                  new-node - nent = temp-s nent;
                    temp - next - perer = new mode;
                     temp-s neut = new-mode;
    Deletion in Doubly Circular linked list:
a)
    At begin :-
     void del- from- begin ()
           Stouct node * temp;
               temp = head;
              if (head ==0)
                     pecints (" list is empty");
                 else if ( head-s next = = head )
                       head = tail = 0;
                         free (temp);
                 else
                        head = head -s nent;
                         head -> prev= tail;
                         tail-s next = head;
                          focus (temp);
   b) At end :-
```

* Same as at begin , only else part changed

```
else
               tail = tail - peer;
               tail - neut = head;
                head -s prev = tail;
                 free (temp);
          3
c) At specified position: -s
          * same code for 'position' before else *
         else
              , while (i < pos)
                     temp = temp = rent;
                       1++ ;
               temp-s prev - s next = temp-s next;
                temp - sneut - s preu = temp - s preu;
               if (temp-s nent = = head)
                    tail = temp - prev;
                    free (temp);
                       free (temp);
                else
    Greation :- s
        Struct mode
             int data;
              Stouct node * nent;
              Stouct node of preen;
         3;
           Struct made & head, & tail *new-node;
```

```
head = 0 int choice = 1;
    while (Choice)
   { new-node = (Struct node *) mella (Size of (Struct node));
       pecinty (" enter data"),
        scanf (".1.d", & new-node-s data);
        y ( head = = 0 )
                head = tail = new-node;
                 head -s neut = head ;
                 head -s priev = head;
           3
         else
                 tail-s next = new-node;
                  new-node - prev= tail,
                   new-mode - nent = head,
                   head -> perer = new-mode;
                    tail = new-node,
              prints (" Do you want to continue: 0 or 1");
               Scanj [". I.d", & Choice) ,
For display
         if (head = =0)
              printy (" list is empty");
          else
                while (temp! = tail)
                    prints (".1.d", temp-solate);
                  3 temp = temp-snext;
                 printy (". I.d", temp -s date),
```