- 1. Write a program to insert and delete a node in linked list.
- > Program to insert and delete a node-

```
# include (staio. h)
# include < stallib.h)
Struct node {
int data:
Struct node * next;
typedy struct node Node;
Node *start, * temp, *p+x, *N;
void create_and_insert ();
void delete_list();
void display ();
 int main ()
   Stust = NULL;
   create_and_invert();
   display ();
   delete - list ();
  return 0;
 wid create-and-invert ()
 int choice;
  do f
    pointf ("Enter the data");
    N= (Node +) malloc (Avze of (Node));
    sconf ("1d", & N-doto);
    91 (Stest == NIVLL)
   -start=temp=N;
```

```
else of
 temponext = N;
  temp=N;
 point ("Press 1 to invest data");
Scanf ("rd", & choice);
? while (choru==1);
temp-next=NULL;
void delete list () {
 int chi
  Printf ("Press 2 to delite data");
 scanf ("1d", & ch);
  if (ch = = 2) {
     temp = start;
     pto = start;
   while (temponext != NULL)
   { pto=temp;
    temp=temp + next;
     ptr -> next = NULL;
    free (temp);
    display ();
  void display () {
  temp=stext;
   printf ("Your list is!");
   while (temp! = NULL) {
   printf (" 1d \ t", temp + data);
   temp = temp - next;
```

(9)

Output:

Enter the data: 10

Press 1 to invest data: 1

Enter the data: 20

Press 1 to invest data: 1

Enter the data: 30

Press 1 to invert data: 2

Your list is: 10 20 30

Press 2 to delete data: 2

Your list is: 10 20

2. Write a program to add two polynomials using linked list.

```
# include (stolio.h)
#include < stallib. h)
type def struct link
  int coeff;
  int pow;
  Struct link * next;
3 my - poly;
void my-create-poly (my-poly * *);
word my- show-poly (my-poly *);
void my-add-poly (my-poly **, my-poly *, my-poly *);
int main (void)
{
int ch;
 dof
 my-poly *poly1 , *poly2 , *poly3;
 printf("In Create 1st expression");
 my-create-poly (&poly 1);
 printf ("Instored the Litexpression");
 my-show-poly(poly1);
 printf ("In (reate 2nd expression");
 my-create-poly(& poly 2);
 pointf ("Instored the 2nd expression");
 my-show-poly(poly2);
 my-add-poly (& poly 3, poly 2, poly 2);
my-show-poly (poly3);
Printf ("In Add two more expression (4=1/N=0:");
scary ("1.4", &ch);
I while (ch);
return o;
```

```
void my-create-poly (my-poly ++ node)
 int flog,
 int coeff, pow;
 my-poly + Improde;
 emp-node = (my-poly *) melloc (size of (my-poly));
 * hode = tmp-node;
  do {
    printf("In Ender Coeff!");
   Seary ("1.2", & coy);
   temp-node - coeff = coeff;
   Printf ("In Enter Pow!")
   scanf (9/1.0", & pow);
  Emp_node -> pow = pow;
   Emp-node→nex+=NULLi
  printf ("In continue adding more terms to the
         polynomial list? (Y=1/N=0):");
  scanf ("".d", & flog);
  if (flog) {
  tmp-node → next= (my-poly *) malloc(size of (my-poly));
  tmp-node = tmp-node - next;
  temp=node → next=NULL;
} while (flag);
void my-show-poly (my-poly * mode)
printf ("In The polynomial expression is: In");
while (node!=null)
  printf ("1.dx" 1.d", node + coeff, node + pow);
  node = hode -> next;
  if (noch != NULL)
     printf("+");
 3
```

```
void my-add-poly (my-poly ** & exult, my-poly * poly 1,
                    my-poly * poly2)
 my-poly *tmp-node;
 tmp-node = (my-poly *) mallor (size of (my-poly));
 Imp-node - next=NULL;
 * result = tmp-node;
  while (poly 1 && poly 2)
     if (poly 1 > pow > poly 2 -> pow)
      Imp-hode > pow = poly 1 + pow;
      tmp-node + coeff = poly 1- coeff;
      poly 1 = poly 1 - next;
     else if (poly 1 -> pow < poly 2 -> pow)
       temp-node - pow = poly 2 + pow;
      temp-node - coeff=poly2 - coeff;
       poly 2 = poly 2 in next;
      else {
       temp-node → pow = poly 1 → pow;
       tmp-node + (veff = poly 1 - coeff + poly 2 - coeff;
       poly 1 = poly1 - next;
      poly 2 = poly 2 - next;
     if ( poly 1. & & poly 2)
       tmp-node - next = (my-poly *) mallor (sizery (my-poly));
      tmp-node = femp-node - next;
       Emp-nocle - next = NULL;
```

```
while (poly 111 poly 2)
tmp-node - next = (my-poly*)malloc (size of (my-poly));
tmp-node=tmp-node-next;
tmp-node-next=NULL;
if (poly 1)
 tmp-node → pow = poly 1 → pow;
 tmp-node-coeff=poly1-coeff;
 poly 1 = poly 1 → next;
if (poly 2)
tmp-node → pow= polyz → pow;
tmp-node - coeff=poly 2 - coeff;
poly 2 = poly 2 - next;
printf ("In Addition complete");
```

Output -

Create 1st expression

Enter (oeff: 7

Continue adding more terms to the polynomial list?(Y=1/N=0): 1

Enter coeff: 6

Enter pow: 2

Continue adding more terms to the polynomial list?(Y=1/N=0): 0

Stored the 1st expression

The polynomial expression is:

7213+6212

Create 2nd Expression

Enter coeffi 3

Enter pow; 3

Confinue adding more terms to the polynomials

list? (Y=1/N=0):1

Enter well: 4

Enter pow: 2

Continue adding more terms to the polynomial list ? (Y=1/N=0):0

Stored the 2nd expression

The polynomial expression is:

3x 13 + 4x12

Addition Complete

The polynomial expression is:

10213 +10212

Add two more expression ! (Y=\$/N=0):0

3. Write a program to insert & delete a node in a circular Linked List.

```
=> Insection Al Beginning
                # include (state f.)
                # include < statib . h)
                 struct node (
                 int infoi
                 Stauct node * hext;
                Struct node * last = NULL;
                void insestAtFront (int data)
                 struct node *temp;
                 temp = (struct node *) malloc (size of (struct node));
                 if (Jast == MULL) {
                  temp - info = data
                  temp + next = temp;
                  Jast = tempi
                  else &
                  temp - info = dato;
                   temp-next=last-next;
                  last - next = temp;
                  void display ()
                  if (last = = NULL)
                      printf ("In List in empty ");
                   else {
                     Struct node *temp;
                      temp= last > next;
                     00
                      prints ("In Data= rd", temp - info);
                     temp = temp - nexti
                     } while (temp!=last → next);
```

```
int main ()
             invest Al Front (10);
              intertal (20);
              (hoot Action (30);
              display();
              defusino;
               Data : 30
                Data: 20
                Data: 10
Insert at end-
                #include (stdio-L)
                #include (stellib.h)
                Struct node {
                  int into i
                  Struct node + next;
                Struct node last = NULL;
                void addutlant (int data)
                 stauct node *temp;
                 temp = (struct nod *) melloc (size of (struct node));
                if (last = = NULL)
                   temp - info = dctc;
                   temp - next = temp;
                   last = temp;
                else
                 temp-info = deta;
                 temp - next = last - next;
                last = next = tempi
                dast = temp;
```

```
void display ()
 if (last == NULL)
    prints ("In List is empty ");
 else &
    Struct node *temp;
    temp = last - next;
    do &
      Printf ("In Dota = 1.d", temp-timo);
   temp = temp - next;
} while (temp! = next (ast - next);
int main ()
 celdatant (10);
  addatlat (20);
  addatlant (30);
 display();
 heturno;
```

Output:

Data: 10 Data: 20 Data: 30

```
Deletion at any point
               # include < state h)
              Hindud (stallib. h)
               struct node {
                 int info;
                 Stouct node * next;
               Stouch rode * dant = MULL;
              void addd last ()
                inf data i
               Stated node * temp;
               temp = (struct node *) melloc (size of (struct nod);
               pointf ("In Enter data to be invested : In");
               searl ("Yd", & deta);
              if (last == NULL) {
                 temp + info = data i
                 temp + next = tempi
                 last = temp;
                else {
                  temp-info=data;
                 temp = next = last +next;
                 last - next = temp;
                 last = temp;
                void deleterationdex ()
                 int pos, i=1;
                 stauct node *temp, *position;
                 temp = last - nexti
                 if (last == NULL)
                   printf ("InList is empty. ");
```

```
else f
 printf ("In Enter index:");
 scanf ("1.d", & por);
 While (ix=pon-1) {
    temp= temp next;
    1++;
  position = temp + next;
  temp - next = position - next;
  free (position);
 woid display()
  if (lust = = NULL)
     printf ("In List is empty h");
  else 5
     struct node *temp;
     temp = last -> nexti
     do {
        pointf("In Data= Y.d", temp -info);
        temp=temp-next;
     fwhile (temp!=lust → next);
 }
   int main ()
    addat last ();
    addat last();
    addatlast();
    delite atindex ();
    display();
   setion ();
```

Output -

Enter data to be inverted:

10

Enter data to be inverted:

20

Enter data to be inverted:

30

Enter index: 2

Data = 30

Data = 30