```
# include <stdio.h>
# include <conio.h>
# include< process.h>
struct node
{ int info;
  struct node* link;
};
typedef struct node *NODE;
NODE getnode()
{NODE x ;
x= (NODE) malloc(sizeof (struct node));
if (x==NULL)
{printf("mem full\n");
 exit(0) ;
}
 return x;
}
void freenode(NODE x)
{free(x) ;
}

NODE insert_rear (NODE first ,int item)
{ NODE temp , cur;
  temp = ~~temp~~getnode();
  temp->info = item;
  temp->link = NULL;
  if (first == NULL){
   return temp;}
   cur = first;
  while (cur->link!= NULL)
   cur = cur->link;
cur -> link = temp;
return first;
}

NODE delete_rear (NODE first)
{
 NODE prev, cur;
 if (first==NULL)
{
 pf(" list is empty");
 return first;
}
 if (first-> link ==NULL)
{
 pf("item deleted is %d", first->info);
 free (first);
 return NULL;
}
 prev= NULL;
 cur=first;
 while (cur -> link != NULL)
{ prev = cur;
  cur =cur -> link;
}
 pf ("item deleted at rear-end is %d", cur->info);
 free(cur);
 prev ->link=NULL;
 return first;
}
```

```
NODE concat (NODE first, NODE second)
{ NODE cur;
 if (first == NULL)
   return second;
 if (second == NULL)
   return first;
 cur = first;
 while (cur -> link != NULL)
 { cur = cur -> link; }
 cur -> link = second;
 return first;
}
```

concaten
joining
at the e
list .
concat
if 2
if on
emp
L1 o
be
i] if L1 is

```
NODE reverse (NODE first)
{
  NODE   cur, temp;
  cur =NULL;
  while (first! = NULL)
 { temp = first;
   first = first -> link ;
   temp -> link = cur ;
   cur = temp;
 }
 return cur;
}
```

```
NODE order_list(int item, NODE first)
temp = getnode();
 temp -> info = item;
  temp -> link = NULL;
  if (first == NULL)
   return temp;
```

```
if (~~temp~~ item < first -> temp)
{
    temp -> link = first;
    return temp;
}
prev = NULL;
cur = first;
while (cur ! = NULL && item > cur-> info)
{   prev = cur;
    cur = cur -> link;
}
 prev -> link = temp;
 temp -> link = cur;
 return first;
}
```

```
void display (NODE first)
{ NODE temp;
  if (first == NULL)
  pf ("list empty");
  for (temp = first ; temp! = NULL ; temp = temp->link)
   { printf ("%d", temp->info);
} }
```

```
void main(){
int item, choice, key, pos;
 int count=0;
NODE first= NULL;
for(;;)
{
printf(" \n 1: Insert_rear\n 2: Delete_rear\n 3: Insert_pos
        \n 4: Delete_pos \n 5: Display \n");
printf(" Enter choice");
Scanf(" %d", & choice);
switch (choice)
{case 1: printf(" Enter item at rear-end");
       Scanf( "%d", & item);
      first = insert_rear (first, item);
        break;
case 2: first = delete_rear(first);
        break;
case 3: printf ("Enter the item to be inserted
                  in ordered list");
   scanf (" %d", &item);
                    order_list (item, first);
   first = ~~delete~~
   break;

case 4: printf ("Enter no.of nodes in 1 \n");
        scanf ("%d", &n);
        a = NULL;
   for(i=0; i<n; i++)
  { printf ("Enter the item\n");
   scanf (" %d", &item);
    a = insert_rear (a, item);
   pf(" Enter the no. of nodes in 2");
   scanf ("%d", & m);
    b=NULL;
   for (i=0; i<n; i++)
  { pf ("Enter item");
    Scanf ("%d", &item);
     b= insert_rear (b, item);
  }
    ~~a= conct (a,b)~~;
     a = concat (a, b);
    display(a);
    break;
case 5 : first = reverse(first);
         display (first);
          break;
 case 4: display (first)
              break;
 default: exit (0) } }
   getch(); }
```