```
typedef struct cs_linked_list
{
  int data;
  struct cs_linked_list  *prv,*nxt;
}CD_NODE;

void insert_in_sorted_order( CD_NODE **aah, int n)
{
   CD_NODE *cur, *prv, *t;
   t= (CD_NODE *)malloc(sizeof(CD_NODE));
   t->data=n;
   p_end=NULL;
   for(cur=*aah, prv=cur->prv; cur!=p_end && n>cur->data;
                                                p_end=*aah, cur=cur->nxt)
            prv=cur;
    if(cur==NULL)
          t->nxt=t;
    else
          t->nxt=cur;
   if(prv)
       prv->nxt=cur;
   else
        *aah=t;
}

void display(C_NODE *ah)
{
  C_NODE *cur, *end_p;
  for(cur=ah, end_p=NULL; cur!=end_p; end_p=ah, cur=cur->nxt)
           printf("\t %d",cur->data);
}
```

```
void deletion(C_NODE **aah, int n)
{
  C_NODE *cur,*prv,*p_end,*t;
  int  found=0;
  if (*aah)
  {
        /*Find the last node pointing to Head Node.*/
        for(cur=(*aah)->nxt; cur->nxt!=*aah; cur=cur->nxt);
       /* ========================================
          Initialisation of prv :
          prv= address of the last node pointing to Head Node.
       ==========================================
       */
        prv=cur;
        p_end=NULL;
        for(cur=*aah; cur!=p_end && found==0; p_end=*aah)
        {
           if(n==cur->data)
           {
              found=1;
              t=cur;
              prv->nxt=cur->nxt;
              free(t);
           }
           else
           {
               prv=cur;
               cur=cur->nxt;
           }
       }
        if(!found)
           printf("\n Item %d not found...",n);
  }
```

```
  else
        printf("\n Empty list…");
}
```