```
#include <iostream.h>
#include <stdlib.h>
#include <stdio.h>
```

/*siano date in ingresso due liste ordinate di numeri interi L1 e L2. Scrivere un programma che, per ogni elemento a della lista L1 verifichi se esiste una coppia di elementi b e c consecutivi nella lista L2 tali che b<a<c. Se tale condizione risulta verificata si elimini a dalla lista L1 e lo si inserisca nella lista L2. Si stampino infine le liste L1 ed L2. */

```
struct nodo
{
 int val;
 nodo* succ;
};

typedef nodo* lista;

 class list
 {
lista l;  // equivale a dire   nodo *l;
  public:
  list () {l=0;} ;
  ~list ();
 bool inserisci (int);
 bool elimina (int);
 void print ();
 void set_farcisce (list l1);
 void farcisce (lista &l2 , list &l1);
 };


 //funzioni membro della classe list
 void list:: set_farcisce (list l1)
 {
 if  ( (l1.l==0)||(l==0)||((l->succ)==0)    )
{
  return;
}

farcisce (l, l1);

   set_farcisce (l1);

 }
```

```
void list:: farcisce (lista &l2 , list &l1)
 {  // l2 sta per l2.l
   int a=l1.l->val;


if ((l2==0)|| (l2->val==0) || (a<(l2->val)) )
{ /* se l1 è vuota, o se l2 ha al più un elemento, o se a<b (essendo la lista ordinata),
   termina la funzione.*/

  return;
 }

  else if ( (a > (l2->val))&& (a < ((l2->succ)->val)) )
{ //se   b<a<c

inserisci (a);  //inserisce a in l2
l1.elimina (a);  //elimina a da l1
return;
}

else
{   farcisce (l2->succ, l1);}

 }  //end farcisce
```

```
 list:: ~list ()     //distruttore
 {
  lista pos;
  printf("Chiamato il distruttore\n");
  while (l!=0)
 {
   pos=l;
   l=l->succ;
   delete pos;
   printf("#");
 }
 printf("\n");

 }// end distruttore
```

```
 bool list::inserisci (int elem)  //forma iterativa
 {
 lista prec, corr, pos;
 if ((pos=new nodo)==0)
{ return false;}

pos->val=elem;
pos->succ=0;
prec=0; //mi posiziono sul primo elemento
corr=l;

while ((corr!=0) && (elem> corr->val))

{ prec=corr;
  corr=corr->succ;
}

if (prec==0)
{
pos->succ=l;
l=pos;
}
else
{
pos->succ=corr;
prec->succ=pos;
}
 return true;
}
```

```
bool list::elimina (int elem)
 {
 lista prec, corr, pos;
 bool continua=true;
 bool trovato=false;
 prec=0;
 corr=l;
 while ( (corr!=0) && (continua) &&(!trovato)  )
{
 trovato =(elem==corr->val);
 if (  continua = (corr->val <elem))
{
  prec=corr;
  corr=corr->succ;
}
```

```
}

if (trovato)
{
 pos=corr;
 if (prec==0)
 l=l->succ;
else
prec->succ=corr->succ;

delete pos;
return true;
}

else
return false;

 } //end elimina


 void list:: print()
 {
 lista pos;
 pos=l;
 while (pos!=0)
{
 cout<<pos->val<<endl;
  pos=pos->succ;
}

 }



void main()   // inizio main
{
list l1, l2;
int elem, scelta, num_lista;

  do
  {
    puts("\nFunzioni disponibili\n");
    puts("\t 1 - Inserisci elemento\n");
    puts("\t 2 - Elimina elemento\n");
    puts("\t 3 - Visualizza lista\n");
    puts("\t 4 - Richiama la funzione farcisce\n");
    puts("\t 5 - Esci\n");
    do
    {
      printf("\rScelta = ");
      if(scanf("%d",&scelta)!=1)
```

```
     {
       fflush(stdin);
            scelta=0;
     }
    } while ( (scelta<1)||(scelta>5));

   switch(scelta)
   {
 case 1:
 printf("\nInserisci un elemento: ");
                  scanf("%d",&elem);
  printf("\nIn quale lista (1/2): ");
                  scanf("%d",&num_lista);
  if (num_lista==1)
{
  if (l1.inserisci (elem))
  printf("\nInserimento effettuato.\n");

   else
  printf("\nInserimento NON effettuato: lista piena\n");

}
 else  if (num_lista==2)
{
  if (l2.inserisci (elem))
  printf("\nInserimento effettuato.\n");

   else
  printf("\nInserimento NON effettuato: lista piena\n");

}

else
{
 printf("Numero di lista errato");
}

    break;

 case 2:
 printf("\nElemento da cancellare: ");
                   scanf("%d",&elem);

   printf("\nIn quale lista (1/2): ");
   scanf("%d",&num_lista);
  if (num_lista==1)
{
  if (l1.elimina (elem))
  printf("\nElemento cancellato.\n");

   else
 printf("\nElemento NON presente.\n");
```

```
}
 else if (num_lista==2)
{
  if (l2.elimina (elem))
  printf("\nElemento cancellato.\n");

   else
 printf("\nElemento NON presente.\n");

}
else
{  printf("Numero di lista errato"); }

 break;

case 3:
  printf("\nQuale lista vuoi stampare (1/2): ");
   scanf("%d",&num_lista);
  if (num_lista==1)
{
  l1.print ();
}
 else if (num_lista==2)
{
  l2.print ();
}

else
{  printf("Numero di lista errato"); }

break;

case 4:

l2.set_farcisce (l1);
cout<<"Adesso le liste appaiono cosi\'\n";
    printf("lista 1:\n");

l1.print ();
 printf("lista 2:\n");
l2.print ();

                break;
    }

  } while(scelta!=5);
       system("PAUSE");
   }
```