实验三 线性表的存储和其它操作

1. The program is as follows:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define TRUE 1
#define FALSE 0
#define OK 1
#define ERROR 0
#define OVERFLOW -2
#define LIST_INIT_SIZE 100
#define LISTINCREMENT 10
typedef int Status;
typedef int ElemType;
typedef struct
{     ElemType *elem;
      int length;
      int listsize;
}SqList;

//Initialize the sequential list
Status InitList_Sq(SqList &L) {
   L.elem = (ElemType *)malloc(LIST_INIT_SIZE*sizeof(ElemType));
   if (!L.elem) return OVERFLOW;
   L.length = 0;
   L.listsize = LIST_INIT_SIZE;
   return OK;
} // InitList_Sq

//Create the sequential list of n elements
Status ListCreate_Sq(SqList &L,int n)
{
      int i;
      srand(time(0));
      for(i=0;i<n;i++)
      {
            L.elem[i] = rand()%90 + 10;
            ++L.length;
      }
      //printf("\n");
      if (L.length==0) return ERROR;
      return OK;
}
```

```
//Output the sequential list
Status ListOutput_Sq(SqList L)
{
      int i;
      if (L.length==0) return ERROR;
      for(i=0;i<L.length;i++)
            printf("%d ",L.elem[i]);
      printf("\n");
      return OK;
}

//Data elements of the sequential list are converse
Status ListConverse_Sq(SqList &L) {
   ElemType temp;
   int i;
   if (L.length==0) return ERROR;
   for(i=0;______________;i++)
   {
      temp=L.elem[i];
      _______________;
      _______________;
   }
   return OK;
} // ListConverse_Sq

void main()
{
      SqList L;
      printf("Initialize the sequential list! ");
      InitList_Sq(L);
      if (L.length==0)
            printf("The sequential list is empty!\n");
      printf("Create the sequential list, ");
      ListCreate_Sq(L,5);
      printf("Output all elements of the sequential list!\n");
      ListOutput_Sq(L);
      ListConverse_Sq(L);
      printf("Output all converse elements of the sequential list\n");
      ListOutput_Sq(L);
}
```

2. The program is as follows:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define TRUE 1
#define FALSE 0
#define OK 1
#define ERROR 0
#define OVERFLOW -2
typedef int Status;
typedef int ElemType;
typedef struct LNode
{     ElemType data;
      struct LNode *next;
}LNode,*LinkList;

//Create the linked list L with n elements from head to rear
void CreateList_L(LinkList &L, int n) {
   LinkList p,q;
   int i;
   L = (LinkList)malloc(sizeof(LNode));
   q=L;
   srand(time(0));
   for (i=1; i<=n; i++) {
      p = (LinkList)malloc(sizeof(LNode));
      p->data = rand()%90 + 10;
      _______________;
      _______________;
   }
   _______________;
} // CreateList_L

//Output the linked list
Status OutputList_L(LinkList L)
{
      LinkList p=L->next;
      if (p==NULL) return ERROR;
      while (_______________)
      {     printf("%d ",p->data);
            _______________;
      }
      printf("\n");
      return OK;
}
```

```
//Data elements of the linked list are converse
Status ListConverse_L(LinkList &L) {
   LinkList p,q;
   p=L->next;
   _______________;
   while (_______________)
   {
         q=p;
         _______________;
         _______________;
         _______________;
   }
   return OK;
} // ListConverse_L

void main()
{
      LinkList L;
      printf("Create the linked list, ");
      CreateList_L(L,5);
      printf("Output all elements of the linked list!\n");
      OutputList_L(L);
      ListConverse_L(L);
      printf("Output all converse elements of the linked list!\n");
      OutputList_L(L);
}
```