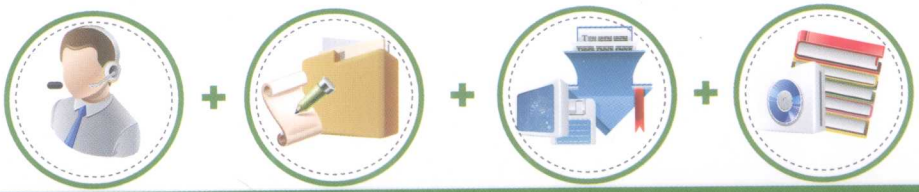
全国高等院校应用型创新规划教材 计算机系列

数据结构

实验指导教程

（C语言版）

李 静 雷小园 主 编
易战军 雷丽兰 陈 军 副主编



- 以理论基础—实用案例—由浅入深的顺序设计为主线
- 紧密结合教材，按照教与学的实际需要取材谋篇
- 精心设置了“数据结构实验”和“数据结构课程设计”两部分内容
- 配备丰富的免费教学资源——本书案例程序的源代码与C++版案例实验教程内容

清华大学出版社

全国高等院校应用型创新规划教材·计算机系列

数据结构实验指导教程(C 语言版)

李　静　雷小园　主　编

易战军　雷丽兰　陈　军　副主编

清华大学出版社

北 京

内 容 简 介

本实验指导教程是配合计算机及相关专业的“数据结构”课程而编写的。在内容编排方面，按照循序渐进、由浅入深的顺序设计、选取案例。全书共分两个部分：第一部分为“数据结构实验”；第二部分为“数据结构课程设计”。

第一部分(包括第 1～8 章)针对每个知识点，首先给出明确的要求，随后设计基础实验，特别是前几章在基础实验之后，设计了若干应用案例。这样有利于学生明确知识点在应用中如何使用，消除迷茫感、增强学习兴趣。

第二部分(即第 9 章)是课程设计，介绍在一个项目中如何选择和使用多种基本的数据结构，介绍如何有效地将它们融合在一起，解决实际的复杂应用问题。

本书可作为高等院校计算机及相关专业数据结构课程的实验教材。

图书在版编目(CIP)数据

数据结构实验指导教程(C 语言版)/李静，雷小园主编. —北京：清华大学出版社，2016
(全国高等院校应用型创新规划教材 • 计算机系列)
ISBN 978-7-302-44860-0

Ⅰ. ①数… Ⅱ. ①李… ②雷… Ⅲ. ①数据结构—高等学校—教材 ②C 语言—程序设计—高等学院—教材 Ⅳ. ①TP311.12 ②TP312.8

中国版本图书馆 CIP 数据核字(2016)第 196550 号

责任编辑：汤涌涛
封面设计：杨玉兰
责任校对：吴春华
责任印制：杨　艳

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn, http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969, c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015, zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn, 010-62791865
印 装 者：三河市春园印刷有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张**：11.25　　**字　数**：282 千字
版　　次：2016 年 9 月第 1 版　　**印　次**：2016 年 9 月第 1 次印刷
印　　数：1～2500
定　　价：28.00 元

产品编号：070252-01

前　言

数据结构是计算机及相关专业中一门重要的专业基础课程。用计算机解决实际问题时，就要涉及数据的表示及数据的处理，而这正是数据结构课程的主要研究对象。通过对数据结构知识内容的学习，可以为后续课程，尤其是软件方面的课程打下坚实的基础，同时，也提供了必要的技能训练。此课程的学习质量将直接影响计算机软件系列课程的学习效果，因此，数据结构课程在计算机专业中具有举足轻重的作用。

根据我们多年的教学经验，认为学生学习数据结构的主要困难在于解题。学生在解题中经常会出现错误，原因在于实践能力不足。

要学好数据结构，仅仅通过课堂教学或自学获取理论知识是远远不够的，还必须加强实际动手能力的训练。只有通过实验课调试和运行已有的各种典型算法和已编写的程序，从成功和失败的经验中得到锻炼，才能熟练掌握和运用理论知识解决软件开发中的实际问题，达到学以致用的目的。

本实验指导教程是配合计算机及相关专业数据结构课程而编写的。本书在内容编排方面，按照循序渐进、由浅入深的顺序设计、选取案例。根据教学内容，针对学生的实际情况，本书在内容编排上共分两个部分。第一部分为“数据结构实验”；第二部分为“数据结构课程设计”。

第一部分(包括第 1～8 章)针对每个知识点，首先给出明确的要求，随后设计基础实验，特别是前几章在基础实验之后，设计了若干应用案例。这样有利于学生明确知识点在应用中如何使用，消除学生的迷茫感、增强学生的学习兴趣。

第二部分(即第 9 章)是课程设计，介绍在一个项目中如何选择和使用多种基本数据结构，介绍如何有效地将它们融合在一起解决实际的复杂应用问题。这有利于学生更深层次地掌握数据结构原理及其应用范围和过程。

本书具有以下特点。

(1) 基于案例驱动的教学内容设计。在实验案例的选择方面，不仅有针对知识点的基础案例，而且有对应的应用案例，从而使学生能够消除畏难情绪。我们在该实验教材的编写过程中，选择案例时由浅入深并精心设计了应用案例，以确保应用的完整性。

(2) 提供大量的源代码和开发案例。在该实验教材的编写中，摒弃了伪代码的描述，全部采用 C 语言源代码，这些源代码都是经过调试并且在教学过程中已经应用的，学生可以直接分析和模仿。同时，在重要的章节，都提供了较为深入的设计案例，例如多项式的运算、括号匹配判断系统、迷宫求解系统、最短路径求解等，为学生提供了更为深入的源码讨论和模仿的机会，极大地提高了教材的全面性、深入性和综合性。

(3) 提供典型的课程设计内容。为了更好地提高学生的专业技能训练水平以及提高学生的学习兴趣，在本书的编写过程中，编写成员根据自己多年教学的积累，整理出了适合

计算机专业学生实际情况的课程设计题目，并提供了相应的解决思路和源代码，为学生提供了很好的学习机会和训练机会。

本书提供案例程序的源代码(可运行)，并赠送 C++版案例实验教程。读者可以从清华大学出版社的网站下载。

本书可作为高等院校计算机及相关专业数据结构课程的实验教材。

由于编者水平有限，错误和不当之处在所难免，希望读者批评指正。

编　者

目录

第 1 章　顺序表 1

实验 1　顺序表的实现 2
1．实验目的 2
2．实验内容 2
3．算法设计 2
4．程序实现 3
5．运行程序 5
实验 2　顺序表的应用——集合运算 5
1．实验目的 5
2．实验内容 5
3．算法设计 5
4．程序实现 6
5．运行程序 8
实验 3　顺序表的应用——回文数猜想 8
1．问题描述 8
2．基本要求 8
3．算法设计 8
4．程序实现 9
5．运行程序 10

第 2 章　链表 11

实验 1　单链表的实现 12
1．实验目的 12
2．实验内容 12
3．算法设计 12
4．程序实现 13
5．运行程序 15
实验 2　单链表的应用——约瑟夫问题 16
1．问题描述 16
2．基本要求 16
3．算法设计 16
4．程序实现 16
5．运行程序 17
实验 3　单链表的应用——多项式求和 18
1．问题描述 18
2．基本要求 18
3．算法设计 18
4．实现程序 18
5．运行程序 21

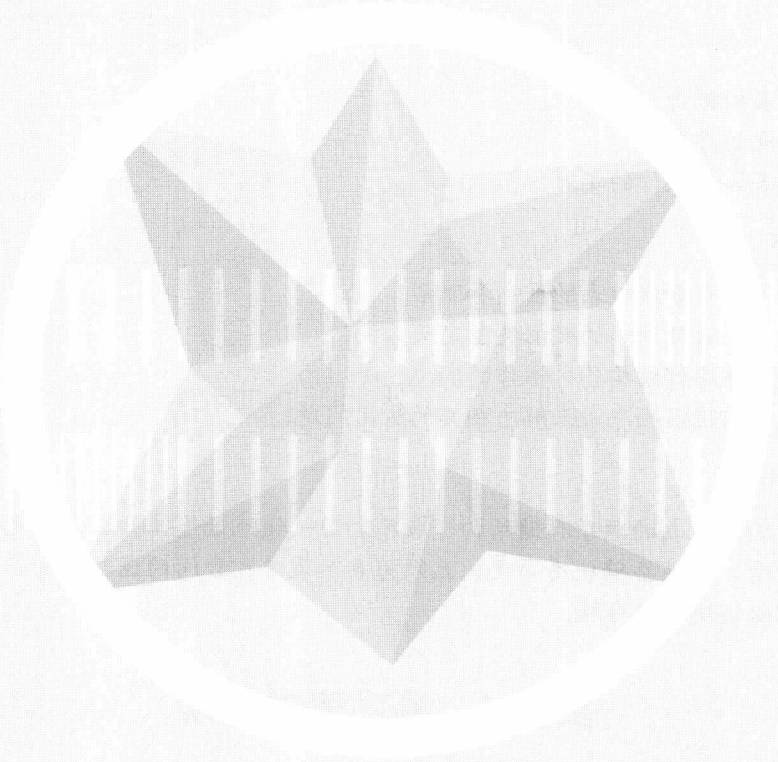
第 3 章　栈 23

实验 1　顺序栈的实现 24
1．实验目的 24
2．实验内容 24
3．算法设计 24
4．程序实现 25
5．运行程序 26
实验 2　链栈的实现 26
1．实验目的 26
2．实验内容 26
3．算法设计 27
4．程序实现 27
5．程序运行 28
实验 3　栈的应用——数制转换 28
1．问题描述 28
2．基本要求 28
3．算法设计 29
4．程序实现 29
5．运行程序 30
实验 4　栈的应用——括号匹配问题 30
1．问题描述 30
2．基本要求 30

3．算法设计.....30
4．程序实现.....30
5．运行程序.....31
实验 5　栈的应用——表达式求值.....32
1．问题描述.....32
2．基本要求.....32
3．算法设计.....32
4．程序实现.....32
5．运行程序.....34
第 4 章　队列.....35
实验 1　循环队列的实现.....36
1．实验目的.....36
2．实验内容.....36
3．算法设计.....36
4．程序实现.....37
5．运行程序.....38
实验 2　链队列的实现.....39
1．实验目的.....39
2．实验内容.....39
3．算法设计.....39
4．程序实现.....39
5．运行程序.....41
实验 3　队列的应用——优先队列.....41
1．问题描述.....41
2．基本要求.....41
3．算法设计.....41
4．实现程序.....42
5．运行程序.....44
实验 4　队列的应用——双端队列.....45
1．问题描述.....45
2．基本要求.....45
3．算法设计.....45
4．程序实现.....45
5．运行程序.....48
第 5 章　二叉树.....49
实验 1　二叉树的建立.....50
1．实验目的.....50
2．实验内容.....50
3．算法设计.....50
4．程序实现.....51
5．运行程序.....51
实验 2　二叉树的遍历.....52
1．实验目的.....52
2．实验内容.....52
3．算法设计.....52
4．程序实现.....53
5．运行程序.....55
实验 3　二叉树的高度、节点数、叶子节点数.....55
1．实验目的.....55
2．实验内容.....55
3．算法设计.....55
4．程序实现.....55
5．运行程序.....57
实验 4　堆.....57
1．问题描述.....57
2．基本要求.....57
3．算法设计.....57
4．程序实现.....58
5．运行程序.....60
第 6 章　图.....61
实验 1　图的邻接矩阵表示.....62
1．实验目的.....62

2．实验内容……62
3．实现提示……62
4．程序实现……62
5．运行程序……64
实验 2　图的邻接表表示……64
1．实验目的……64
2．实验内容……64
3．实现提示……64
4．程序实现……64
5．运行程序……66
实验 3　图的深度优先搜索……67
1．问题描述……67
2．基本要求……67
3．实现提示……67
4．程序实现……67
5．运行程序……69
第 7 章　排序……71
实验 1　冒泡排序……72
1．实验目的……72
2．实验内容……72
3．实现提示……72
4．程序实现……73
5．运行程序……74
实验 2　插入排序、选择排序……74
1．实验目的……74
2．实验内容……74
3．实现提示……75
4．程序实现……75
5．运行程序……76
实验 3　归并排序……76
1．实验目的……76
2．实验内容……76
3．实现提示……76
4．实现程序……76
5．运行程序……78
实验 4　快速排序……78
1．实验目的……78
2．实验内容……79
3．实现提示……79
4．程序实现……79
5．运行程序……80
实验 5　堆排序……81
1．实验目的……81
2．实验内容……81
3．实现提示……81
4．程序实现……81
5．运行程序……82
第 8 章　查找……83
实验 1　折半查找……84
1．实验目的……84
2．实验内容……84
3．实现提示……84
4．程序实现……85
5．运行程序……86
实验 2　二叉排序树查找……87
1．实验目的……87
2．实验内容……87
3．实现提示……87
4．程序实现……87
5．运行程序……89
实验 3　哈希查找……89
1．实验目的……89
2．实验内容……89
3．实现提示……90

4．程序实现……90
5．运行程序……91
第 9 章　课程设计……93
问题 1　学生成绩管理……94
1．问题描述……94
2．任务要求……94
3．程序实现……95
4．运行结果……98
问题 2　数据库管理系统……98
1．问题描述……98
2．任务要求……98
3．分析与实现……99
4．程序实现……101
5．运行结果……116
问题 3　马踏棋盘……117
1．问题描述……117
2．任务要求……117
3．分析与实现……117
4．运行结果……120
问题 4　停车场管理……121
1．问题描述……121
2．任务要求……121
3．分析与实现……122
4．运行结果……126
问题 5　大整数计算器……126
1．问题描述……126
2．任务要求……127
3．分析与实现……127
4．运行结果……132
问题 6　魔方阵……132
1．问题描述……132
2．任务要求……133
3．分析与实现……133
4．运行结果……134
问题 7　本科生导师制问题……134
1．问题描述……134
2．任务要求……135
3．分析与实现……135
4．运行结果……144
问题 8　电文的编码和译码……145
1．问题描述……145
2．任务要求……145
3．分析与实现……145
4．运行结果……148
问题 9　家族关系查询系统……149
1．问题描述……149
2．任务要求……149
3．分析与实现……149
4．运行结果……161
问题 10　地铁建设问题……162
1．问题描述……162
2．任务要求……162
3．分析与实现……162
4．运行结果……165
问题 11　校园导航……165
1．问题描述……165
2．任务要求……165
3．分析与实现……166
4．运行结果……169
参考文献……170



第1章

顺序表

本章要点

(1) 顺序表的概念。
(2) 顺序表的存储。
(3) 顺序表各种操作的实现。

学习目标

(1) 理解顺序表和线性表的区别和联系。
(2) 掌握顺序存储结构的数据类型定义方法。
(3) 掌握顺序存储结构各种操作的实现。
(4) 掌握如何使用顺序表来解决相关的应用问题。

基本知识点

顺序表是指线性表的顺序存储结构，顺序表用一组地址连续的存储单元依次存放线性表中的数据元素。顺序存储使用简便、无须为表示表中元素间的逻辑关系而额外增加存储空间，并且可以实现随机存取。

实验 1　顺序表的实现

1. 实验目的

(1) 掌握顺序表的存储结构。
(2) 验证顺序表及其基本操作的实现。
(3) 理解算法与程序的关系，能够将顺序表算法转换为对应的程序。

2. 实验内容

(1) 初始化顺序表。
(2) 在顺序表的第 i 位插入元素。
(3) 删除顺序表的第 i 个元素。
(4) 输出顺序表。
(5) 判断顺序表是否为空。
(6) 判断顺序表是否满。
(7) 求顺序表第 i 个元素的值。
(8) 查找值为 x 的元素。

3. 算法设计

用结构体来描述顺序表，结构体中包括表的大小、存放数据的数组、表的最大容量三个数据属性。

为简单起见，本实验假定线性表的数据元素为 int 型。

结构体的定义如下：

```
typedef int dataType;
typedef struct {
    dataType *data;
    int size;
    int maxSize;
} SqList;
```

应实现顺序表的初始化、插入、删除、判空、判满、求值、查找、输出等操作。

(1) void InitList(SqList *l)：初始化顺序表。

(2) void Insert(SqList *l, int k, dataType x)：在顺序表 l 的第 k 个位置插入元素 x。

(3) void Delete(SqList *l, int k)：删除顺序表 l 的第 k 个元素。

(4) int Empty(SqList *l)：判断顺序表是否为空。

(5) int Full(SqList *l)：判断顺序表是否满。

(6) dataType GetData(SqList *l, int i)：求顺序表 l 中第 i 个元素的值。

(7) int locate(SqList *l, dataType x)：在顺序表 l 中查找值为 x 的元素。

(8) void Print(SqList *l)：输出顺序表。

4．程序实现

程序完整的实现代码如下：

```
#include <stdio.h>
#include <malloc.h>
#define INIT_SIZE 100
typedef int dataType;
typedef struct {
    dataType *data;
    int size;
    int maxSize;
} SqList;

//初始化顺序表
void InitList(SqList *l) {
    l->data = (dataType*)malloc(INIT_SIZE * sizeof(dataType));
    l->size = 0;
    l->maxSize = INIT_SIZE;
}

//在顺序表 l 的第 k 个位置插入元素 x
void Insert(SqList *l, int k, dataType x) {
    if (k<1 || k>l->size+1) exit(1);
    if (l->size == l->maxSize) exit(1);
    for (int i=l->size; i>=k; i--)
        l->data[i] = l->data[i-1];
    l->data[k-1] = x;
    l->size++;
}
```

```
//删除顺序表 l 的第 k 个元素
void Delete(SqList *l, int k) {
    if (k<1 || k>l->size) exit(1);
    for (int i=k; i<l->size; i++)
        l->data[i-1] = l->data[i];
    l->size--;
}

//判断顺序表是否为空
int Empty(SqList *l) {
    return l->size == 0;
}

//判断顺序表是否满
int Full(SqList *l) {
    return l->size == l->maxSize;
}

//求顺序表 l 中第 i 个元素的值
dataType GetData(SqList *l, int i) {
    if (i<1 || i>l->size) exit(1);
    return l->data[i-1];
}

//在顺序表 l 中查找值为 x 的元素
int locate(SqList *l, dataType x) {
    for (int i=0; i<l->size; i++)
        if (l->data[i] == x)
            return i + 1;
    return 0;
}

//输出顺序表
void Print(SqList *l) {
    for (int i=0; i<l->size; i++)
        printf("%d ", l->data[i]);
    printf("\n");
}

int main() {
    SqList list, *pList=&list;
    InitList(pList);
    Insert(pList, 1, 10);
    Insert(pList, 1, 20);
    Delete(pList, 2);
    Insert(pList, 1, 30);
    Insert(pList, 1, 40);
    Print(pList);
    printf("%d", GetData(pList, 2));
}
```

5．运行程序

运行程序后，将会显示如图 1.1 所示的界面。

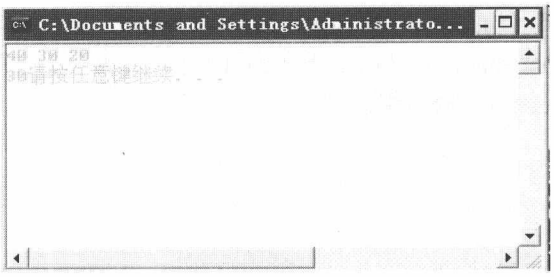


图 1.1　顺序表的输出

实验 2　顺序表的应用——集合运算

1．实验目的

(1) 学会使用顺序表来解决集合运算问题。
(2) 掌握集合的交、并运算的实现。

2．实验内容

(1) 创建一个空的集合。
(2) 从数组元素建立集合。
(3) 查找集合中是否存在元素 x。
(4) 往集合中增加元素 x。
(5) 删除集合的元素 x。
(6) 求两个集合的并集。
(7) 求两个集合的交集。
(8) 输出集合。

3．算法设计

用顺序表来存储集合，结构体中包含集合的大小、存放集合的数组。
为简单起见，本实验假定集合中的数据元素为 char 型。结构体的定义如下：

```
typedef char dataType;
typedef struct {
    dataType *data;
    int size;
} SqSet;
```

应实现集合的并和交运算等操作。

(1) SqSet* CreateSet()：创建一个空的集合。
(2) SqSet* CreateSetFromArray(dataType a[], int n)：从数组元素建立集合。

(3) int Find(SqSet *s, dataType x)：查找集合中是否存在元素 x。

(4) void Add(SqSet *s, dataType x)：往集合中增加元素 x。

(5) void Delete(SqSet *s, dataType x)：删除集合中的元素 x。

(6) void Union(SqSet *a, SqSet *b, SqSet *c)：求两个集合的并集。

(7) void Intersection(SqSet *a, SqSet *b, SqSet *c)：求两个集合的交集。

(8) void Print(SqSet *s)：输出集合。

4. 程序实现

程序完整的实现代码如下：

```
#include <stdio.h>
#include <malloc.h>
#define MaxSize 100

typedef char dataType;
typedef struct {
    dataType data[MaxSize];
    int size;
} SqSet;

//创建一个空的集合
SqSet* CreateSet() {
    SqSet *t = (SqSet*)malloc(sizeof(SqSet));
    t->size = 0;
    return t;
}

//从数组元素建立集合
SqSet* CreateSetFromArray(dataType a[], int n) {
    SqSet *t = (SqSet*)malloc(sizeof(SqSet));
    t->size = n;
    for (int i=0; i<n; i++)
        t->data[i] = a[i];
    return t;
}

//查找集合中是否存在元素 x
int Find(SqSet *s, dataType x) {
    for (int i=0; i<s->size; i++)
        if (s->data[i] == x)
            return 1;
    return 0;
}

//往集合中增加元素 x
void Add(SqSet *s, dataType x) {
    if (Find(s, x)) return;
    s->data[s->size] = x;
    s->size++;
```

```
//删除集合中的元素 x
void Delete(SqSet *s, dataType x) {
    for (int i=0; i<s->size; i++) {
        if (s->data[i] == x) {
            s->data[i] = s->data[--s->size];
            return;
        }
    }
}

//求两个集合的并集
void Union(SqSet *a, SqSet *b, SqSet *c) {
    for (int i=0; i<a->size; i++)
        c->data[i] = a->data[i];
    c->size = a->size;
    for (int i=0; i<b->size; i++)
        if (!Find(c, b->data[i]))
            c->data[c->size++] = b->data[i];
}

//求两个集合的交集
void Intersection(SqSet *a, SqSet *b, SqSet *c) {
    c->size = 0;
    for (int i=0; i<a->size; i++)
        if (Find(b, a->data[i]))
            c->data[c->size++] = a->data[i];
}

//输出集合
void Print(SqSet *s) {
    for (int i=0; i<s->size; i++)
        printf("%c ", s->data[i]);
    printf("\n");
}

int main() {
    dataType a[] = {'a', 'c', 'e', 'h'};
    dataType b[] = {'f', 'h', 'b', 'g', 'd', 'a'};
    SqSet *s1, *s2, *s3;
    s1 = CreateSetFromArray(a, 4);
    s2 = CreateSetFromArray(b, 6);
    s3 = CreateSet();
    printf("第一个集合的元素为：");
    Print(s1);
    printf("第二个集合的元素为：");
    Print(s2);
    Union(s1, s2, s3);
    printf("两个集合的并为：");
```

```
    Print(s3);
    Intersection(s1, s2, s3);
    printf("两个集合的交为: ");
    Print(s3);
}
```

5. 运行程序

运行程序后，将会显示如图 1.2 所示的界面。

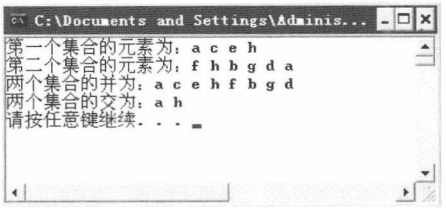


图 1.2　集合的交和并

实验 3　顺序表的应用——回文数猜想

1. 问题描述

一个正整数，如果从左向右读(称为正序数)和从右向左读(称为倒序数)是一样的，这样的数就叫回文数。任取一个正整数，如果不是回文数，将该数与它的倒序数相加，若其和不是回文数，则重复上述步骤，一直到获得回文数为止。例如：68 变成 154(68+86)，再变成 605(154+451)，最后变成 1111(605+506)，而 1111 是回文数。

于是有数学家提出一个猜想：不论开始是什么正整数，在经过有限次正序数和倒序数相加的步骤后，都会得到一个回文数。至今为止还不知道这个猜想是对还是错。现在请你编程验证。

2. 基本要求

(1) 使用顺序表来存储运算过程中产生的正整数，最后对这个数据进行显示。

(2) 假如输入一个数 27228，则输出 27228→109500→115401→219912。

(3) 假如输入 37649，则输出 37649→132322→355553。

3. 算法设计

(1) 先写一个简单的顺序表，实现顺序表的如下两个功能：

```
SqList* createSqList();                    //创建顺序表
void push(SqList *list, dataType x);       //在顺序表后面增加一个元素
```

(2) 写一个从一个正整数得到其倒序数的函数 reverse()。

(3) 将输入的整数存放到顺序表中。

(4) 将这个整数与其倒序数比较，看是否相等。如果不等，则将这个倒序数存放到顺序表的后面。