实验 0　数组、指针和结构体

题目一 数据集合的表示及运算

【问题描述】

构造一个非递减数列，然后在该数列中新加入一个数，并保持该数列非递减有序的特性。

【基本要求】

用一维数组存储数列。

【实现提示】

1. 为简单起见，设数组元素为整型，用静态数组（即固定大小的数组）存储数列，定义如下：

```
#define    ARRAYSIZE    10000
//先定义一个足够大的数组，并存入少量数据测试程序的逻辑是否正确
int data[ARRAYSIZE] = {10,20,30,40,50};
int n = 5;     //  数列的长度用 n 表示，程序中应保证 n 不大于 ARRAYSIZE
```

2. 新加入的数从键盘输入，该数有以下几种可能性，在运行程序时都要进行测试：

（1）比数列中最小的数还要小；

（2）比数列中最大的数还要大；

（3）其他情况

3. 调用随机函数产生数列及要加入的新数，这就不能保证数列的非递减性质了，因此需要在程序中增加一个专门用来排序的函数；

4*. 用动态数组，即用动态内存申请函数申请内存块（作为数组的存储空间），借助指针访问数组元素，数组大小自行设定，例如：

```
#define    DSIZE    100
int n = 5;     //数列的长度用 n 表示，程序中应保证 n 不大于 DSIZE
int *ptr;
ptr = (int *)malloc(DSIZE * sizeof(int));
//calloc 也可以用来申请内存块
//与 malloc 不同的是，calloc 函数会将申请到的内存区域初始化为 0
//用法示例：ptr = (int *)calloc(DSIZE, sizeof(int));
if (!ptr)         //申请失败，程序中止
    exit(0);
for(i = 0; i<n; i++)
    *(ptr+i) = rand();       //调用随机函数产生数据，*(ptr+i)等同于 ptr[i]
```

【测试数据】

自行设定。

题目二 约瑟夫问题

【问题描述】

约瑟夫问题是由公元 1 世纪时的犹太历史学家（也是军官及辩论家）弗拉维奥·约瑟夫斯提出的，他参加并记录了公园 66-70 年犹太人反抗罗马的起义。他和他的 40 个战友被罗马军队包围在洞

中。他们在讨论是自杀还是被俘，最终决定自杀，并以抽签的方式决定谁杀掉谁。约瑟夫斯和另外一个人是最后两个留下的人。约瑟夫斯说服了那个人，他们将向罗马军队投降，不再自杀。约瑟夫斯把他的存活归因于运气或天意，他不知道是哪一个。

在计算机编程的算法中，类似问题又称为**约瑟夫环**：n 个人（编号 1~n）围成一圈，从编号为 1 的人开始，依顺时针方向进行。从 1 开始报数，报到 m 的人退出，从下一个人开始，继续从 1 开始报数，报到 m 的人退出，重复上述过程，直到仅剩 1 人为止（称为胜利者）。按顺序输出出列者编号，以及胜利者的编号。

【基本要求】

用一维数组存储。

【测试数据】

m 和 n 的值自行设定。

题目三 复数运算

【问题描述】

复数是复变函数论、解析数论、傅里叶分析、分形、流体力学、相对论、量子力学等学科中最基础的对象和工具。设 z1=a+bi，z2=c+di 是任意两个复数，则复数的四则运算为：

1. 相加 z3 = z1 + z2：z3 = a+bi + c+di = (a+c) + (b+d)i
2. 相减 z3 = z1 + z2：z3 = a+bi –(c+di)= (a-c) + (b-d)i
3. 相乘 z3 = z1 * z2：z3 = (a+bi) * (c+di) = (ac-bd)+(bc+ad)i
4. 相除 z3 = z1 / z2：z3 = (a+bi) / (c+di) = $\frac{ac+bd}{c^2+\mathrm{d}^2}+\frac{bc-ad}{c^2+d^2}i$

【基本要求】

1. 用结构体类型描述复数数据；
2. 定义四个函数分别实现复数的四则运算;
3. 实部和虚部为 double 型，输出时小数点后保留 2 位。

【测试数据】

自行设定。

实验一 链表的实现及运算

题目一 单链表基本运算

【问题描述】

设计并实现线性表的单链表存储和运算。

【基本要求】

实现单链表的插入、删除和遍历运算，每种操作用一个函数实现。

插入操作：将一个新元素插入表中指定序号的位置。

删除操作：将指定序号的元素从表中删除。

遍历操作：从表头按次序输出所有元素的值，若是空表，则输出信息“empty list!”。

【实现提示】

1. 程序运行时，首先在 main 函数中创建空的、带头结点的单链表。然后多次调用实现插入操作

的函数（每次都将元素在序号 1 位置上插入），将元素依次插入表中，最后调用实现遍历操作的函数输出所有元素。之后再多次调用实现删除操作的函数将表还原为空表（每次都删除第 1 个元素，每删除一个元素后，将表中剩余元素都输出一次）。

2. 单链表结点类型定义：

```
typedef int ElemType;    //为简化起见，元素类型定义为整型
typedef struct Node{
      ElemType data;
      struct Node *next;
}Node, *LinkList;
```

3. 为了简化指针参数传递，使用 c++的引用参数，存储**源程序**时注意后缀名应为 **cpp**。（其他实验题目在此处的处理相同）

4．创建链表时，可以多次调用插入函数（插入函数的功能是在链表的第 i 个元素结点前插入一个新结点），通过不断地在链表中增加结点来实现，也可以定义一个专门创建链表的函数，调用一次该函数就完成链表中所有结点的创建（当然，此后需要在链表中插入一个新结点时调用前述的插入函数即可）。

例如，用头插法创建长度为 n 的单链表的函数定义如下：

```
Status    CreateList_L(LinkList &L, int n)
{     //用表头插入法逆序建立长度为 n 的、带头结点的单链表
      int i;
      LinkList p;
      L = (LinkList) malloc(sizeof(Node));
      if (!L)
         return INFEASIBLE;
      L->next = NULL;      //建立仅有头结点的单链表

      for(i=n; i>0; --i) {     //输入 n 个整数，逐个插入单链表中
            p = (LinkList)malloc(sizeof(Node));      //生成新结点
            if (!p)
                return INFEASIBLE;
            scanf("%d",&p->data);                    //从键盘输入元素值，存入新结点中
            p->next = L->next;    L->next = p;       //新结点插入到表头
      }
      return OK;
}
```

【测试数据】

输入数据：1　2　3　4　5　0（为 0 时结束，0 不存入链表）

第一次输出：5　4　3　2　1

第二次输出：4　3　2　1

第三次输出：3　2　1

第四次输出：2　1

第五次输出：1

第六次输出：empty list!

题目二 单链表上的排序运算

【问题描述】

创建单链表，用冒泡排序方法（或其他排序方法）实现非递减排序。

【基本要求】

创建单链表，遍历单链表并输出元素序列，将单链表中的元素按非递减顺序排序，再次遍历单链表并输出元素序列。

【实现提示】

分别定义函数实现创建单链表、遍历单链表并输出和排序过程。程序运行时，首先在 main 函数中调用创建单链表的函数，然后调用遍历单链表（包含对元素的输出）的函数，再调用排序函数，最后再调用遍历单链表（包含对元素的输出）的函数观察是否实现了排序运算。

【测试数据】

自行设计，输入的数据序列应该是无序的。

题目三 约瑟夫问题

【问题描述】

编号为 1,2,...,n 的 n 个人按顺时针方向围坐一圈，每人持有一个密码（正整数）。现在给定一个随机数 m>0，从编号为 1 的人开始，按顺时针方向 1 开始顺序报数，报到 m 时停止。报 m 的人出圈，同时留下他的密码作为新的 m 值，从他在顺时针方向上的下一个人开始，重新从 1 开始报数，如此下去，直至所有的人全部出列为止。

【基本要求】

利用单向循环链表存储结构模拟此过程，按照出列的顺序打印输出每个人的编号。

【测试数据】

M 的初始值为 20；n 等于 7，7 个人的密码依次为：3，1，7，2，4，8，4。

输出为：6，1，4，7，2，3，5

【实现提示】

程序运行时，首先要求用户指定初始报数上限值，然后读取各人的密码。可设 n≤30。此题所用的循环链表中不需要“头结点”，请注意空表和非空表的界限。

【选作内容】

用顺序存储结构实现该题目。

题目四 一元多项式相加、减运算器

【问题描述】

设计一个一元稀疏多项式简单计算器。

【基本要求】

一元稀疏多项式简单计算器的基本功能：

（1）输入并建立多项式的单向链表存储；

（2）输出多项式，形式为一个整数序列：$n,c_1,e_1,c_2,e_2,\cdots,c_n,e_n$，其中，n 是多项式的项数，$c_i$、$e_i$ 分别是第 i 项的系数和指数，序列按指数降序排列；

（3）多项式 A(x)和 B(x)相加得到多项式 C(x)，输出 C(x)；

（4）多项式 A(x)和 B(x)相减得到多项式 D(x)，输出 D(x)。

【测试数据】

（1）$A(x)=2x+5x^8-3.1x^{11}\quad B(x)=7-5x^8+11x^9\quad C(x)=-3.1x^{11}+11x^9+2x+7$

（2）$A(x)=1+x+x^2+x^3+x^4+x^5\quad B(x)=-x^3-x^4\quad C(x)=1+x+x^2+x^5$

（3）$A(x)=x+x^3\quad B(x)=-x-x^3\quad C(x)=0$

（4）$A(x)=x+x^{100}\quad B(x)=x^{100}+x^{200}\quad C(x)=x+2x^{100}+x^{200}$

（5）$A(x)=x+x^2+x^3\quad B(x)=0\quad C(x)=x+x^2+x^3$

（6）$A(x)=6x^{-3}-x+4.4x^2-1.2x^9\quad B(x)=-6x^{-3}+5.4x^2-x^2+7.8x^{15}$
$D(x)=-7.8x^{15}-1.2x^9+12x^{-3}-x$

【实现提示】

用带头结点的单链表存储多项式，多项式的项数可放入头结点中。

实验二 栈和队列的实现与应用

题目一 数制转换

【问题描述】

十进制数 N 和其它 d 进制数的转换是计算机实现计算的基本问题，其解决方法很多，其中一个简单的算法是基于下列原理：

N = (N div d) * d + N mod d。其中：div 为整除运算，mod 为求余运算。

例如，$43_{10}=101011_2$

被除数	除数	商	余数
43	2	21	1
21	2	10	1
10	2	5	0
5	2	2	1
2	2	1	0
1	2	0	1

从上述运算过程可见：①从低位到高位顺序产生二进制数的各个位数；②与打印输出由高位到低位的顺序相反。

因此，将计算过程中得到的二八进制数的各位顺序进栈，再按出栈序列打印输出，即可得到与输入对应的二进制数。

【基本要求】

写一个程序，利用栈将一个十进制数转换为二进制数的形式输出，其中，栈的基本运算应自行实