云南大学数学与统计学实验教学中心  
《高级语言程序设计》实验报告

|  |  |  |
| --- | --- | --- |
| 课程名称：程序设计和算法语言 | 学期：2016~2017学年上学期 | 成绩： |
| 指导教师：赵越 | 学生姓名：刘鹏 | 学生学号：20151910042 |
| 实验名称：数组程序设计 |  |  |
| 实验编号：No.05 | 实验日期：2018年8月16日 | 实验学时：2 |
| 学院：数学与统计学院 | 专业：信息与计算科学 | 年级：2015级 |

# 实验目的

1. 掌握数组的概念和使用方法。
2. 掌握数组初始化的方法。
3. 学会字符数组和字符串的应用。
4. 学会用数组名作函数的参数。
5. 掌握一维数组与二维数组的定义及其元素的引用方法。
6. 深刻体会数组与循环的关系。
7. 掌握利用一维数组和二维数组实现一些常用算法的编程技巧。
8. 进一步掌握动态调试的基本技能。

# 实验环境

Windows10 Pro Workstation 17134.165；

Cygwin GCC编译器。

# 实验内容

## 有关概念

* + - 1. 只有静态数组和外部数组才能初始化。

1. 引用数组时，编译器对下标是否越界不作检查。如定义int a[5];在引用时出现a[5];不给出错信息，而是引a[4]下面一个单元的值。
2. 字符串放在字符数组中，一个字符串以'\0'结束，有一些字符串函数如strcpy，strcmp，strlen等可以方便进行字符串运算。
3. 如有如下定义：char \*str = "I love china";表示str是一个字符型指针变量，它的值是一个字符数据的地址。不要认为str是字符串变量，在其中存放一个字串"I love china"。
4. 用数组名作函数实参时，传到形参的是数组的首地址。

## 题1

定义三个数组

int a**[**5**];**

int b**[**2**][**2**];**

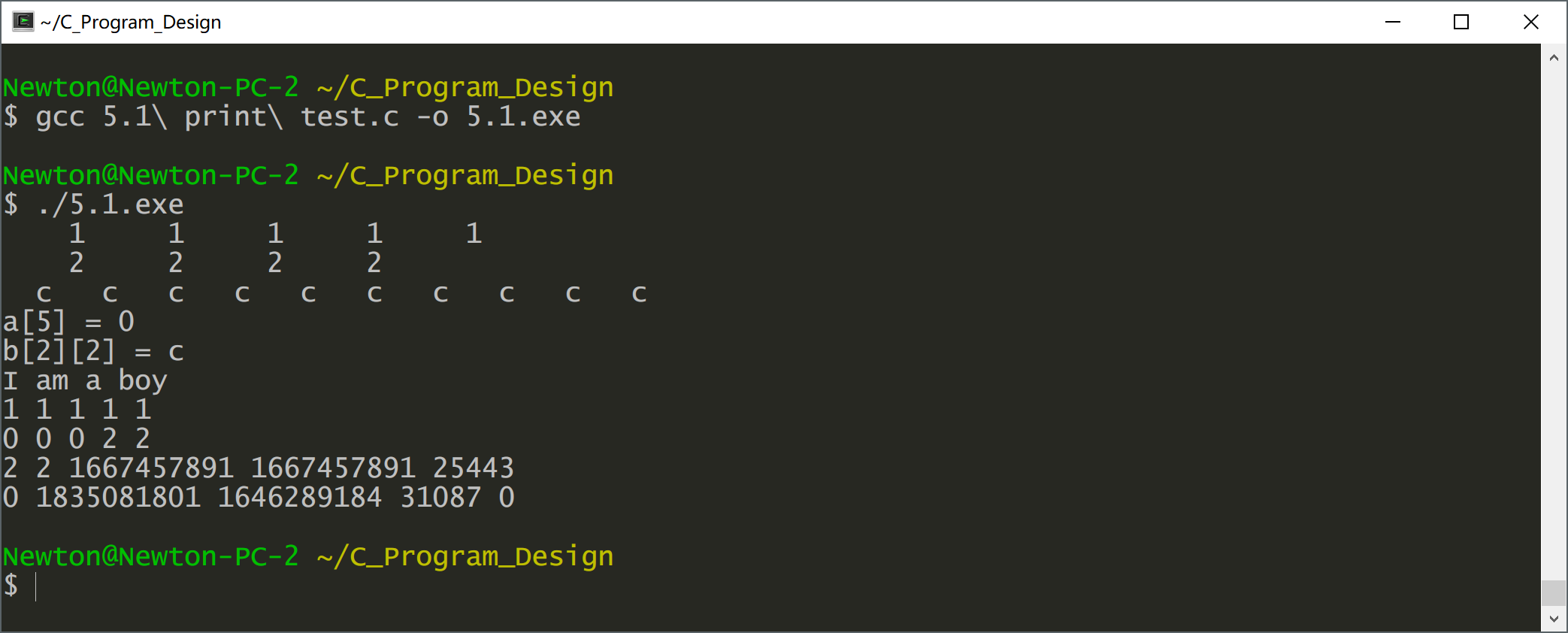
char c**[**10**];**

分别在函数体外和函数体内对它们进行初始化，然后输出它们的值。在程序中再加一语句，输出a**[**5**],** b**[**2**][**2**]**，分析结果。对c数组改为用赋值语句给各元素赋初值：c[0]～c[9]各元素分别为：'I'**,** ' '**,** 'a'**,** 'm'**,** ' '**,** 'b'**,** 'o'**,** 'y'。然后用printf**(**"%s"**,** c**)**输出字符串，分析结果。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39 | /\*  \* filename: 5.1 print test.c  \* property: test  \*/  #include <stdio.h>  int main**()** **{**  int i**,** x**,** y**;**  static int a**[**5**]** **=** **{**1**,** 1**,** 1**,** 1**,** 1**};**  static int b**[**2**][**2**]={{**2**,** 2**},** **{**2**,** 2**}};**  static char c**[**10**]** **=** **{**'c'**,** 'c'**,** 'c'**,** 'c'**,** 'c'**,** 'c'**,** 'c'**,** 'c'**,** 'c'**,** 'c'**};**  **for(**i **=** 0**;** i **<** 5**;** i**++)**  printf**(** "%5d "**,** a**[**i**]);**  printf**(**"\n"**);**  **for(**x **=** 0**;** x **<** 2**;** x**++)**  **for(**y **=** 0**;** y **<** 2**;** y**++)**  printf**(**"%5d "**,**b**[**x**][**y**]);**  printf**(**"\n"**);**  **for(**i **=** 0**;** i **<** 10**;** i**++)**  printf**(**"%3c "**,** c**[**i**]);**  printf**(**"\n"**);**  printf**(**"a[5] = %d\n"**,** a**[**5**]);**  printf**(**"b[2][2] = %c\n"**,** b**[**2**][**2**]);**  static char c2**[**10**]** **=** "I am a boy"**;**  printf**(**"%s\n"**,** c2**);**    **for(**int i**=**0**;** i **<** 20**;** i**++)** **{**  printf**(**"%d "**,** a**[**i**]);**  **if((**i**+**1**)** **%** 5 **==** 0**)** **{**  printf**(**"\n"**);**  **}**  **}**  **return** 0**;**  **}** |

### 运行结果



## 搜索

有一数组，内放10个整数，要求找出最小的数和它的下标。然后把它和数组中最前面的元素对换位置。

编写程序，上机运行，并记录下结果。提示：数组的个元素可用输入函数scanf()通过键盘输入进去，找出数组中最小的元素可通过循环语句和条件语句来实现。

设min是存放数组中最小元素的变量，array[k]为一个暂存单元。实现最前面的元素与最小元素对换可通过下面语句实现：

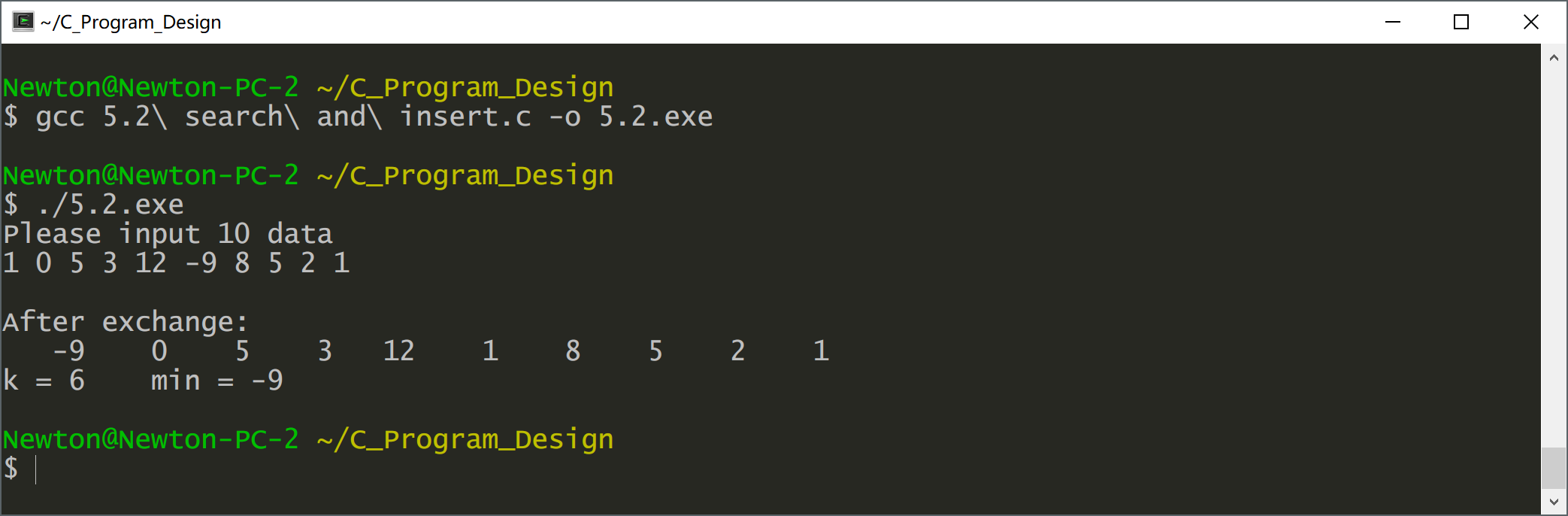
array**[**k**]=**array**[**0**];**

array**[**0**]=**min**;**

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | /\*  \* filename: 5.2 search and insert.c  \* property: test  \*/  #include <stdio.h>  int main**()** **{**  int i**;**  int array**[**10**];**  int min**,** k **=** 0**;**  printf**(**"Please input 10 data\n"**);**  **for(**i **=** 0**;** i **<** 10**;** i**++)**  scanf**(**"%d"**,** **&**array**[**i**]);**  min **=** array**[**0**];**  printf**(**"\n"**);**  **for(**i **=** 1**;** i **<** 10**;** i**++)**  **if(**min **>** array**[**i**])** **{**  min **=** array**[**i**];**  k **=** i**;**  **}**  array**[**k**]** **=** array**[**0**];**  array**[**0**]** **=** min**;**  printf**(**"After exchange:\n"**);**  **for(**i **=** 0**;** i **<** 10**;** i**++)**  printf**(**"%5d"**,** array**[**i**]);**  printf**(**"\n"**);**  printf**(**"k = %d\t min = %d\n"**,** k **+** 1**,** min**);**  **return** 0**;**  **}** |

### 运行结果



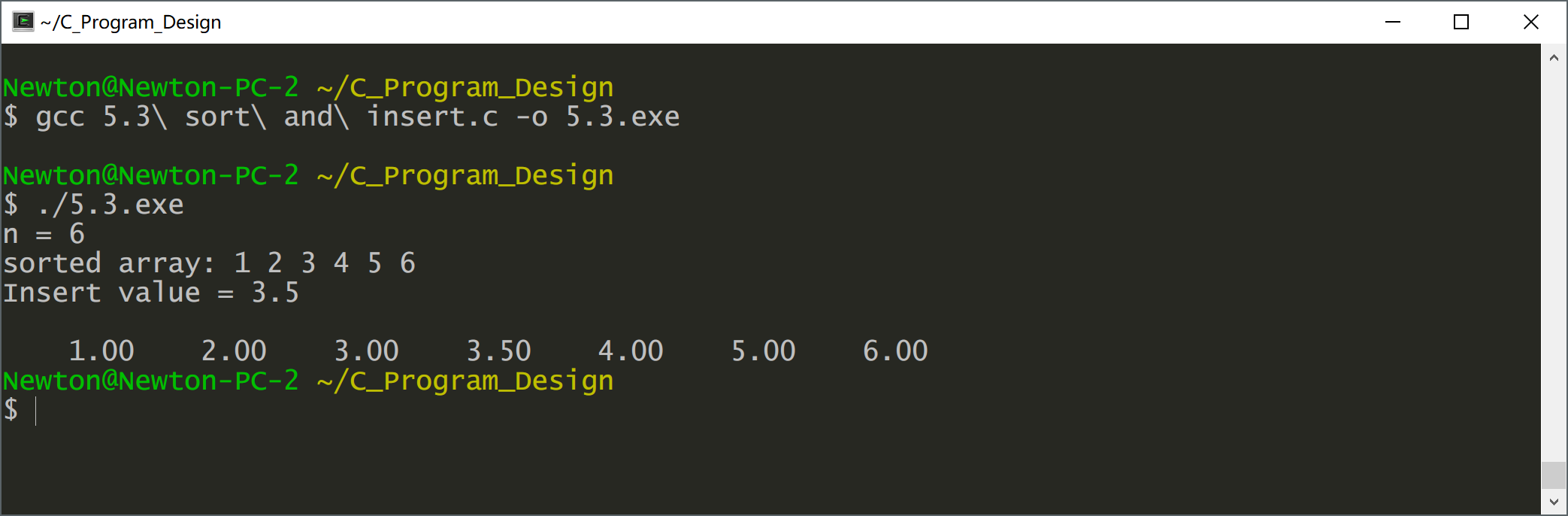
## 有序插入

在一个已排好序的数列中（由小到大）再插入一个数，要求仍然有序。编程并上机运行。提示：编程时应考虑到插入的数的各种可能性（比原有所有的数大；比原有所有的数小；在最大数和最小数之间）。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | /\*  \* filename: 5.3 sort and insert.c  \* property: exercise  \*/  #include <stdio.h>  int main**()** **{**  int i**,** n**;**  float a**,** x**[**20**],** y**[**21**];**  printf**(**"n = "**);**  scanf**(**"%d"**,** **&**n**);**  printf**(**"sorted array: "**);**  **for(**i **=** 0**;** i **<** n**;** i**++)**  scanf**(**"%f"**,** **&**x**[**i**]);**  printf**(**"Insert value = "**);**  scanf**(**"%f"**,** **&**a**);**  i **=** 0**;**  **while(**a **>** x**[**i**]** **&&** i **<** n**)** **{**  y**[**i**]** **=** x**[**i**];**  i**++;**  **}**  y**[**i**]** **=** a**;**  **for(**i **=** i**+**1**;** i **<** n **+** 1**;** i**++)**  y**[**i**]** **=** x**[**i**-**1**];**  printf**(**"\n"**);**  **for(**i**=**0**;**i**<**n**+**1**;**i**++)** **{**  printf**(**"%8.2f"**,** y**[**i**]);**  **}**  **return** 0**;**  **}** |

### 运行结果



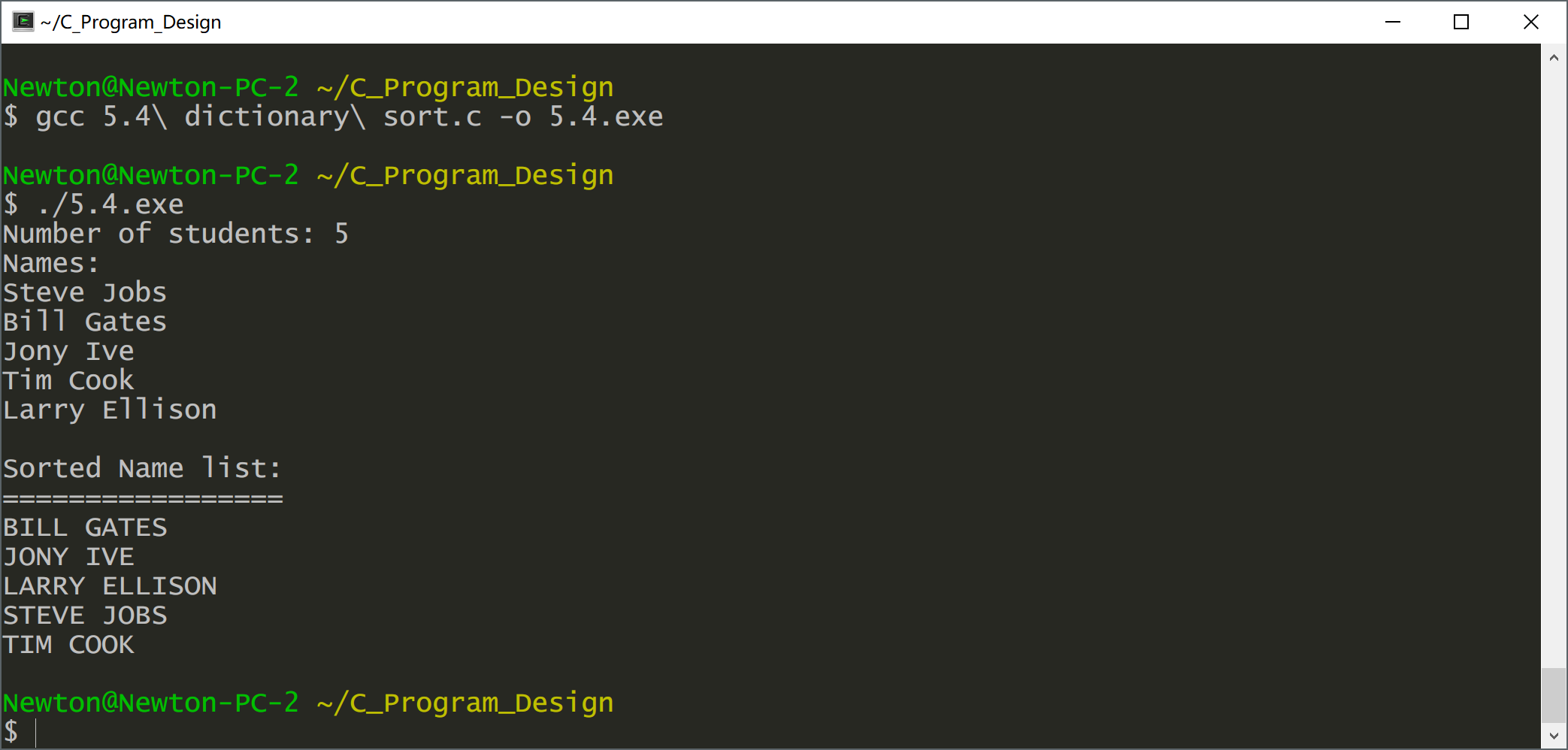
## 按姓名排序

编写一程序，一班级有名学生要求按他们姓名的顺序排列（按汉语拼音的字母顺序从小到大），并按序输出。

### 程序代码

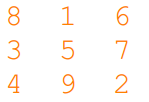
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45 | /\*  \* filename: 5.4 dictionary sort.c  \* property: exercise  \*/  #include <stdio.h>  #include <string.h>  void strup**(**char str**[])** **{**  int i**;**  **for(**i**=**0**;** str**[**i**]** **!=** '\0'**;** i**++)**  **if(**str**[**i**]** **>=** 'a' **&&** str**[**i**]** **<=** 'z'**)**  str**[**i**]** **=** str**[**i**]** **+** 'A' **-** 'a'**;**  **}**  int main**()** **{**  char name**[**20**][**20**];**  int i**,** j**,** n**;**  printf**(**"Number of students: "**);**  scanf**(**"%d"**,** **&**n**);**  n **+=** 1**;**  printf**(**"Names:\n"**);**  **for(**i **=** 0**;** i **<** n**;** i**++)** **{**  gets**(**name**[**i**]);**  strup**(**name**[**i**]);**  **}**  char tmp**[**20**];**  // small -> big, bubble sort  **for** **(**i **=** 0**;** i **<** n **-** 1**;** i**++)** **{**  **for** **(**j **=** i**;** j **<** n**;** j**++)** **{**  **if(**strcmp**(**name**[**i**],** name**[**j**])** **>** 0**)** **{**  strcpy**(**tmp**,** name**[**i**]);**  strcpy**(**name**[**i**],** name**[**j**]);**  strcpy**(**name**[**j**],** tmp**);**  **}**  **}**  **}**    printf**(**"\nSorted Name list:\n================="**);**  **for(**i **=** 0**;** i **<** n**;** i**++)**  printf**(**"%s\n"**,** name**[**i**]);**  **return** 0**;**  **}** |

### 运行结果



## \* 打印魔方阵

所谓魔方阵是指，它的每行每一列的和与对角线之和均相等。例如，三阶魔方阵为



要求打印由到的自然数构成的魔方阵。

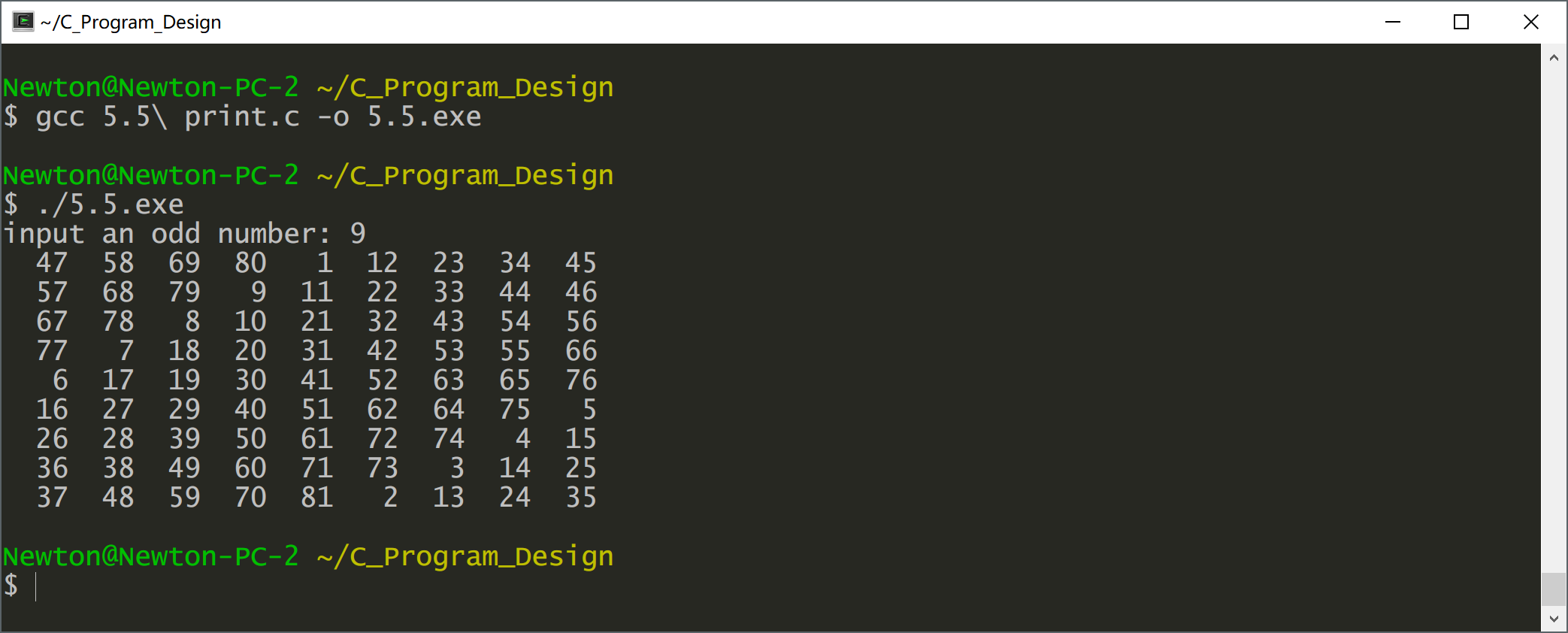
提示：魔方阵中各数排列规律为：将“1”放在第一行中间一列；从“2”开始直到，各数依次按下列规则存放：每一个数存放的行比前一个数的行数减1，列数加1；如果上一数的行数为1，则下一个数的行数应为（指最下一行）；当上一个数的列数为时，下一个数的列数应为1，行数减1。

如果按上面的规则，确定的位置上已有数，或上一个数是第1行第列时，则把下一个数放在上一个数的下面。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45 | /\*  \* filename: 5.5 print.c  \* property: exercise  \*/  #include <stdio.h>  #define N 10  int main**()** **{**  int n**,** i**,** j**,** a**,** b**,** num**,** u**,** v**;**  int c**[**10**][**10**];**  printf**(**"input an odd number: "**);**  scanf**(**"%d"**,** **&**n**);**  num **=** n **\*** n**;**  **for(**i **=** 0**;** i **<** n**;** i**++)**  **for(**j **=** 0**;** j **<** n**;** j**++)**  c**[**i**][**j**]** **=** 0**;**  c**[**0**][**n**/**2**]** **=** 1**;**  a **=** 0**;**  b **=** n**/**2**;**  **for(**i **=** 2**;** i **<=** num**;** i**++)** **{**  **if(**a**)**  u **=** a **-** 1**;**  **else**  u **=** n **-** 1**;**    v **=** **(**b **+** 1**)** **%** n**;**  **if(!**c**[**u**][**v**])**  c**[**u**][**v**]** **=** i**;**  **else** **{**  u **=** **(**a **+** 1**)** **%** n**;**  v **=** b**;**  c**[**u**][**v**]** **=** i**;**  **}**  a **=** u**;**  b **=** v**;**  **}**  **for(**i **=** 0**;** i **<** n**;** i**++)** **{**  **for(**j **=** 0**;** j **<** n**;** j**++)**  printf**(**"%4d"**,** c**[**i**][**j**]);**  printf**(**"\n"**);**  **}**  **return** 0**;**  **}** |

### 运行结果



## 元素移位

用移位法将数组a中的最后一个数移到最前面，其余数依次往后移动一个位置。

请按以下步骤实习和思考：

①分析程序及其特性。

②上机运行程序，查看运行结果是否正确？

③用动态跟踪查找错误原因，以此分析并找出错误原因。

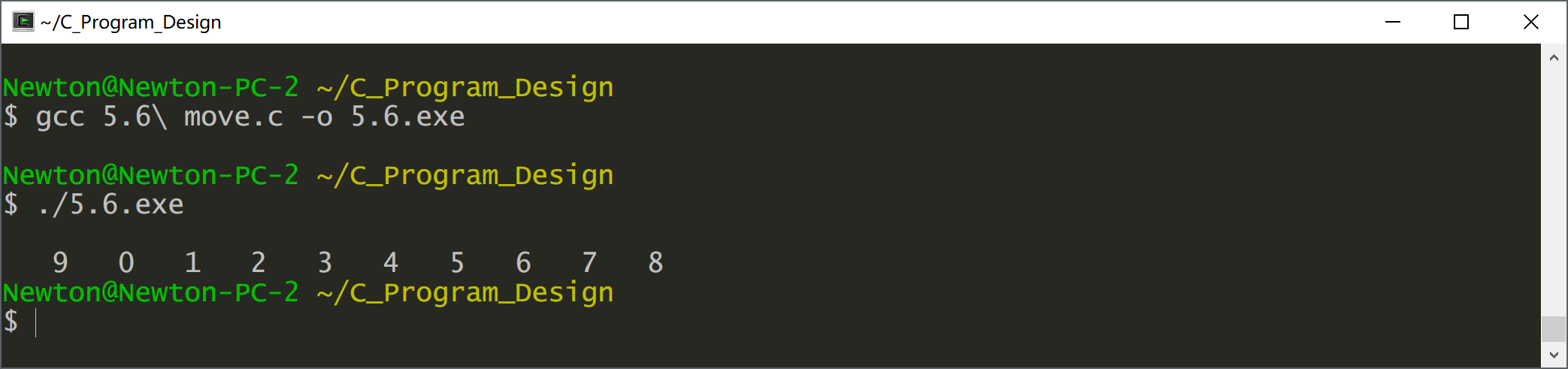
④改正错误后重新运行程序，直到结果正确为此。

⑤如果要用三次循环移位来实现将最后三个数移到前面，其余数依次往后移三个位置，则程序应该如何修改？

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | /\*  \* filename: 5.6 move.c  \* property: test  \*/  #include <stdio.h>  int main**()** **{**  int i**,** t**;**  int a**[**10**]** **=** **{**0**,** 1**,** 2**,** 3**,** 4**,** 5**,** 6**,** 7**,** 8**,** 9**};**  t **=** a**[**9**];**  **for(**i **=** 9**;** i **>** 0**;** i**--)**  a**[**i**]** **=** a**[**i**-**1**];**  a**[**0**]** **=** t**;**  printf**(**"\n"**);**  **for(**i **=** 0**;** i **<** 10**;** i**++)**  printf**(**"%4d"**,** a**[**i**]);**  **return** 0**;**  **}** |

### 运行结果



## 成绩排序

输入个学生的单科成绩，然后从高到低的顺序排序后输出。

①分析程序及其特性。

②上机编译程序，程序是否有语法错误？应如何修改？（数组a的长度可比n大些），改正错误后重新编译和运行程序，直到结果正确为此。

③你对选择排序算法的实现过程是否清楚了？若不清楚，请用动态跟踪的方法观察其实现过程，操作如下：首先将光标移至 if语句行上，按F4，接着输入数据，当绿条第停留在if语句行时，用Ctrl-F7操作将a数组的内容显示出来，不断按F4，观察a数组值的变化情况，以此分析和领会算法的实现过程。

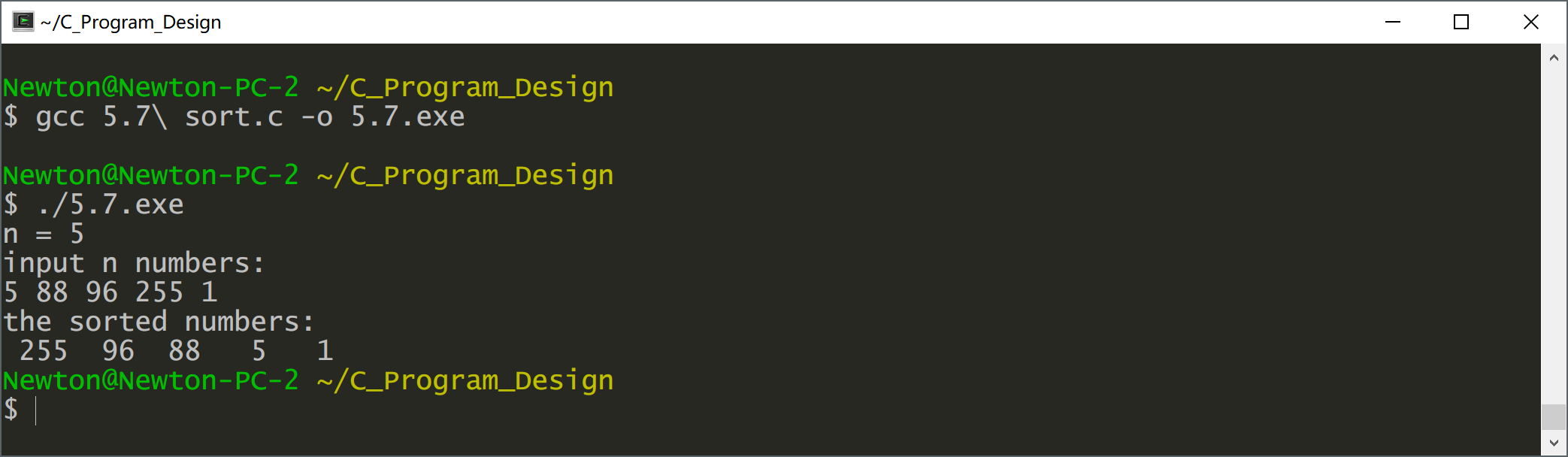
④输入冒泡排序程序，用动态跟踪观察其实现过程。

⑤如果要用三次循环移位来实现将最后三个数移到前面，其余数依次往后移三个位置，则程序应该如何修改？

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | /\*  \* filename: 5.7 sort.c  \* property: test  \*/  #include <stdio.h>  #define N 10  int main**()** **{**  int i**,** j**,** t**,** n**,** a**[**N**];**  printf**(**"n = "**);**  scanf**(**"%d"**,** **&**n**);**  printf**(**"input n numbers: \n"**);**  **for** **(**i **=** 0**;** i **<** n**;** i**++)**  scanf**(**"%d"**,&**a**[**i**]);**  **for(**i **=** 0**;** i **<** n **-** 1**;** i**++)**  **for(**j **=** i **+** 1**;** j **<** n**;** j**++)**  **if(**a**[**i**]** **<** a**[**j**])** **{**  t **=** a**[**i**];**  a**[**i**]** **=** a**[**j**];**  a**[**j**]** **=** t**;**  **}**  printf**(**"the sorted numbers:\n"**);**  **for(**i **=** 0**;** i **<** n**;** i**++)**  printf**(**"%4d"**,** a**[**i**]);**  **return** 0**;**  **}** |

### 运行结果



## 矩阵操作

将矩阵Ａ[4][5]中值（行中所有数的和）为最大的那一行元素与首行元素对换。

具体要求如下：

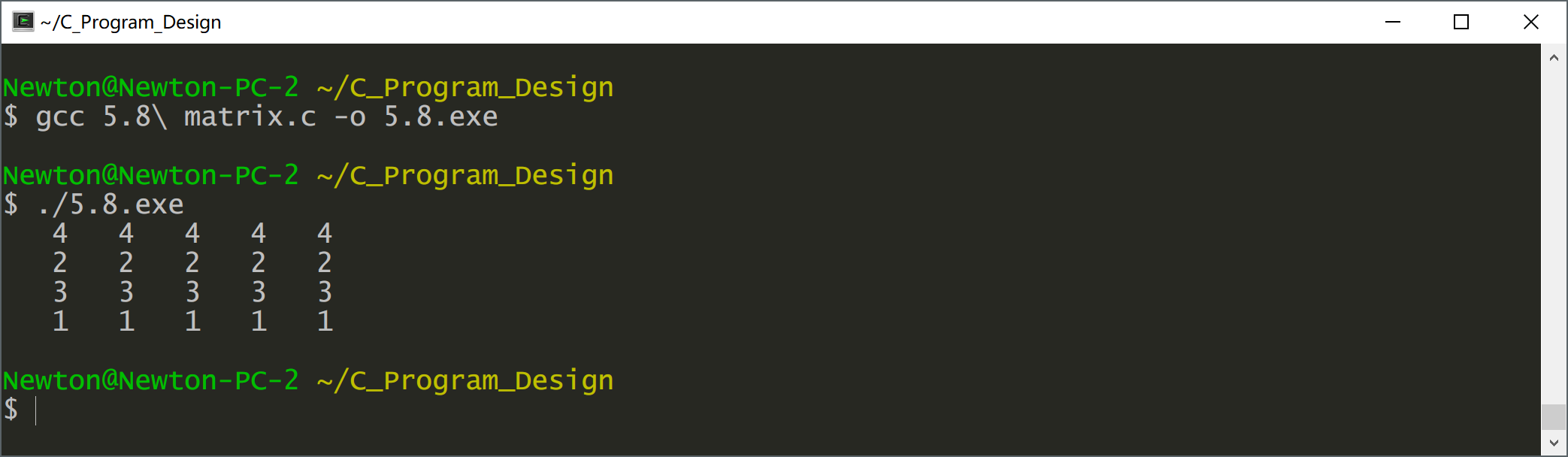
①矩阵Ａ的数值从键盘输入。（可以采取固定内容实例）

②以矩阵的形式输出对换后的矩阵。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | /\*  \* filename: 5.8 matrix.c  \* property: exercise  \*/  #include <stdio.h>  int main**()** **{**  int i**,** j**;**  int a**[**4**][**5**]** **=** **{{**1**,** 1**,** 1**,** 1**,** 1**},** **{**2**,** 2**,** 2**,** 2**,** 2**},** **{**3**,** 3**,** 3**,** 3**,** 3**},** **{**4**,** 4**,** 4**,** 4**,** 4**}};**  int d**[**4**]** **=** **{**0**};**  **for** **(**i **=** 0**;** i **<** 4**;** i**++)** **{**  **for** **(**j **=** 0**;** j **<** 5**;** j**++)** **{**  d**[**i**]** **+=** a**[**i**][**j**];**  **}**  **}**  int max\_index**;**  max\_index **=** **(**d**[**0**]** **>** d**[**1**])?** 1**:**2**;**  max\_index **=** **(**max\_index **>** d**[**2**])?** max\_index**:**3**;**  max\_index **=** **(**max\_index **>** d**[**3**])?** max\_index**:**4**;**    int tmp**;**  **if** **(**max\_index **!=** 1**)** **{**  **for** **(**i **=** 0**;** i **<** 5**;** i**++)** **{**  tmp **=** a**[**0**][**i**];**  a**[**0**][**i**]** **=** a**[**max\_index**-**1**][**i**];**  a**[**max\_index**-**1**][**i**]** **=** tmp**;**  **}**  **}**  **for** **(**i **=** 0**;** i **<** 4**;** i**++)** **{**  **for** **(**j **=** 0**;** j **<** 5**;** j**++)** **{**  printf**(**"%4d"**,** a**[**i**][**j**]);**  **}**  printf**(**"\n"**);**  **}**  **return** 0**;**  **}** |

### 运行结果

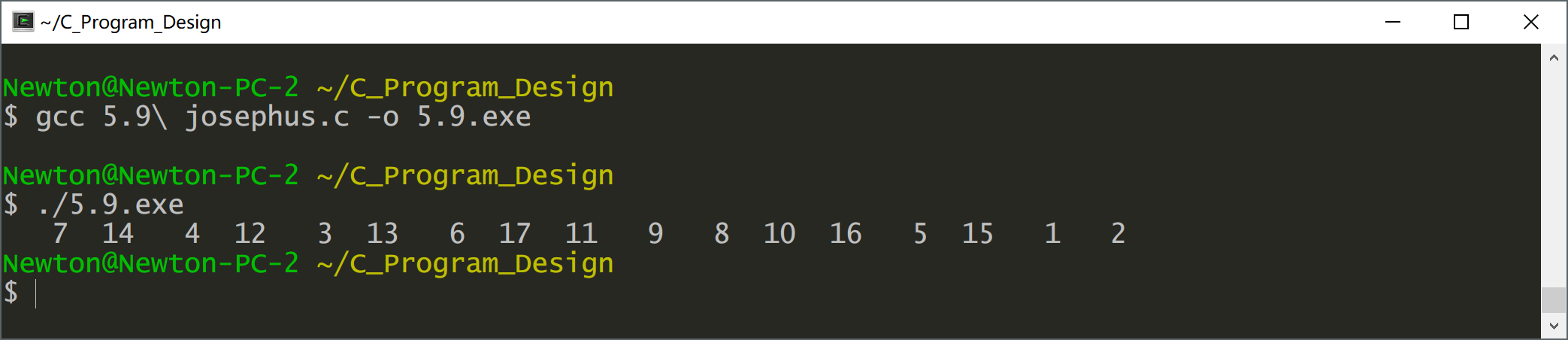


17个人围坐一圈，顺序编号为1，2，3，…，17。现在从第一个人开始数起，每数到7时，这个人就从圈里出来，再从下一个数重新开始数1，2，…，7，数到第7的这个人也从圈里出来，直到全部17个人从圈里出来为此。例如，前面站出来的4个人是7，14，4和12。编程输出从圈里出来的人的顺序。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45 | /\*  \* filename: 5.9 josephus.c  \* property: exercise  \*/  #include <stdio.h>  int main**()** **{**  int i**,** j **=** 0**;**  int a**[**17**];** // 0~16 == 1~17  **for** **(**i **=** 0**;** i **<** 17**;** i**++)** **{**  a**[**i**]** **=** 1**;**  **}**  int order**[**17**];**  int count **=** 0**;**  int left **=** 16**;**  int next **=** 0**;**  **while** **(**left **>=** 0**)** **{**  **while** **(**count **<** 7**)** **{**  **if** **(**a**[**next**]** **==** 0**)** **{**  next **=** **(**next **+** 1**)** **%** 17**;**  **}**  **else** **{**  **if** **(**count **<** 6**)**  next **=** **(**next **+** 1**)** **%** 17**;**  count **+=** 1**;**  **}**  **}**  order**[**j**]** **=** next**;**  a**[**next**]** **=** 0**;**  next **=** **(**next **+** 1**)** **%** 17**;**  count **=** 0**;**  left**--;**  j**++;**  **}**  **for** **(**i **=** 0**;** i **<** 17**;** i**++)** **{**  printf**(**"%4d"**,** order**[**i**]+**1**);**  **}**    **return** 0**;**  **}** |

### 运行结果



## 程序排错

下列有关数组的说明或语句是否存在错误？

int a**[**3**,**4**];** a**[**3**]=**a**[**1**\***2**]; // 错误，初始化错误**

int a**[**3**]={**10**,**8**,**3**,**4**} // 错误，超界初始化**

float a**[**4**];**a**[**0**]=**15**;**a**[**4**]=**100**; // 错误，超界**

int I**,**j**,**k**[**3**]={**3**}; // 正确**

char ch**[]={}; // 正确**

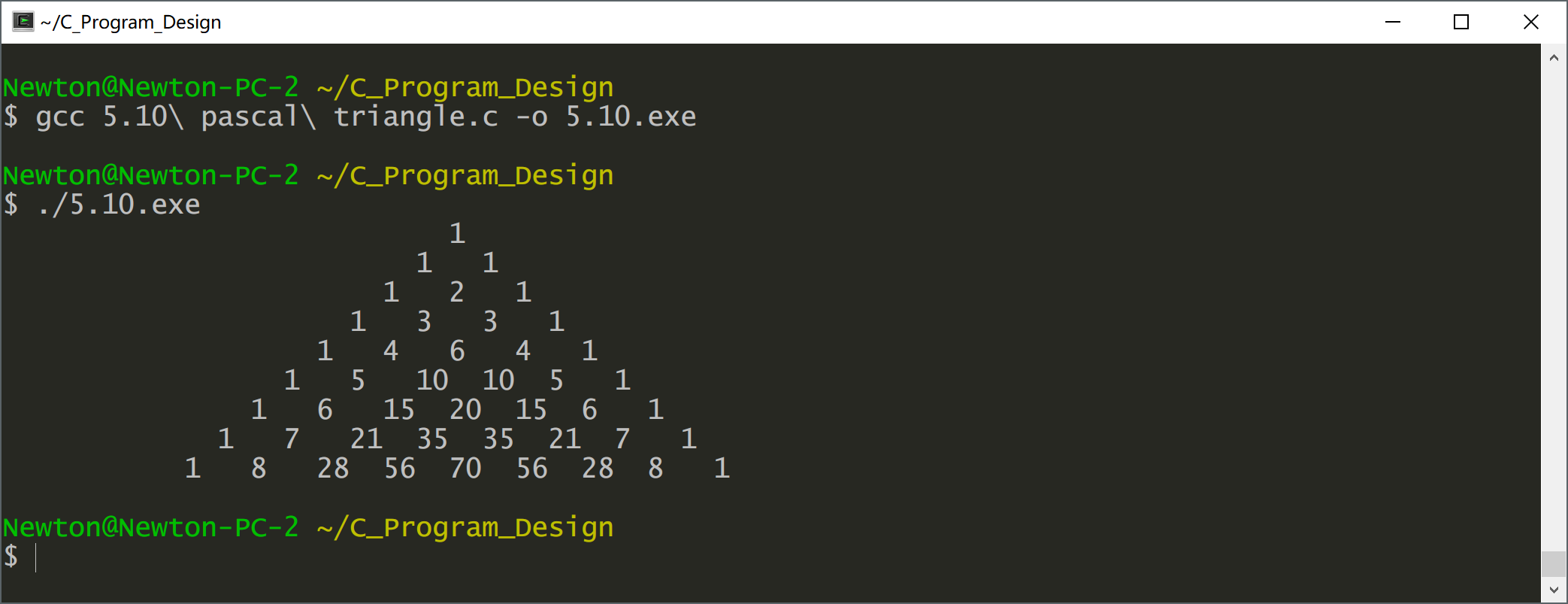
## 输出杨辉三角

写一个输出10阶杨辉三角形的程序。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | /\*  \* filename: 5.10 pascal triangle.c  \* property: test  \*/  #include <stdio.h>  #define N 10  int main**()** **{**  int a**[**N**][**N**],** i**,** j**;**  **for(**i **=** 1**;** i **<** N**;** i**++)** **{**  a**[**i**][**1**]** **=** 1**;**  a**[**i**][**i**]** **=** 1**;**  **}**  **for(**i **=** 3**;** i **<** N**;** i**++)**  **for(**j **=** 2**;** j **<=** i **-** 1**;** j**++)**  a**[**i**][**j**]** **=** a**[**i**-**1**][**j**-**1**]** **+** a**[**i**-**1**][**j**];**  **for(**i **=** 1**;** i **<** N**;** i**++)** **{**  **for(**j **=** 1**;** j **<** 30 **-** 2 **\*** i**;** j**++)**  printf**(**" "**);**  **for(**j **=** 1**;** j **<=** i**;** j**++)**  printf**(**"%-4d"**,**a**[**i**][**j**]);**  printf**(**"\n"**);**  **}**    **return** 0**;**  **}** |

### 运行结果



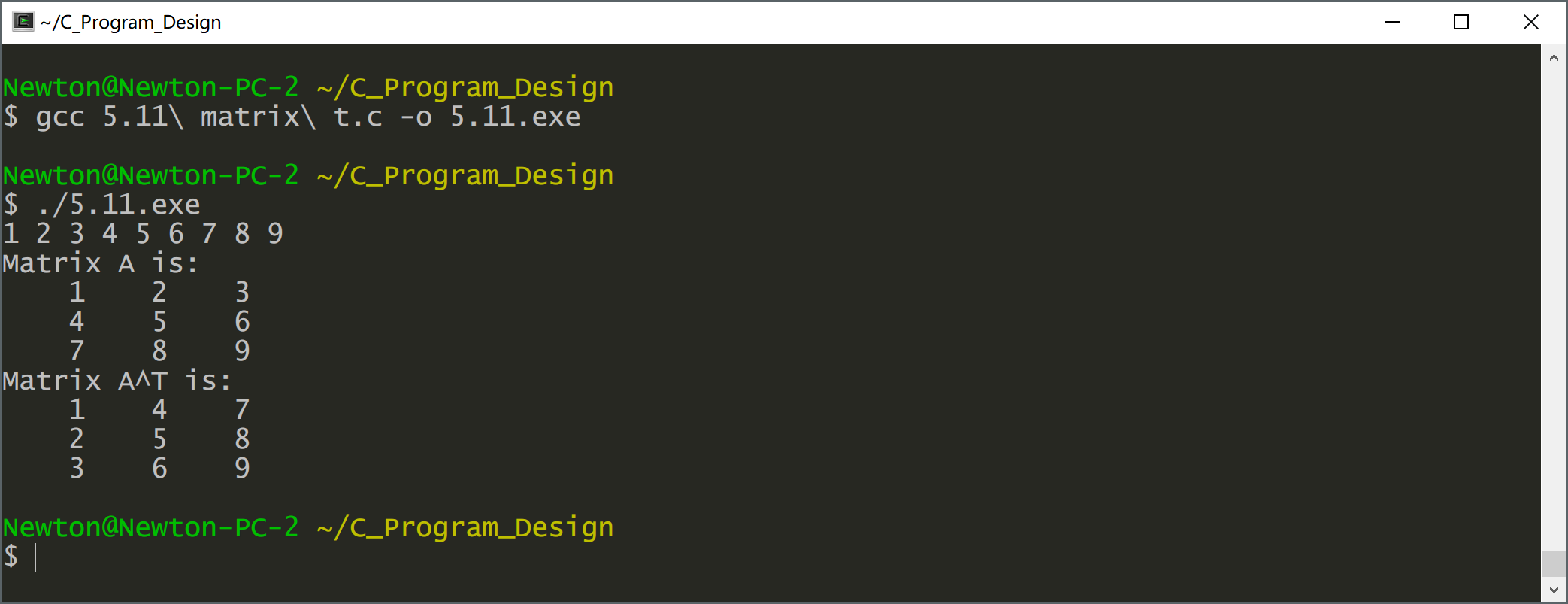
## 矩阵转置

从键盘输入的矩阵（可以定义为5），输出此矩阵和转置后的矩阵。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | /\*  \* filename: 5.11 matrix t.c  \* property: test  \*/  #include <stdio.h>  #define N 3  int main**()** **{**  int a**[**N**][**N**],** b**[**N**][**N**],** i**,** j**;**  **for(**i**=**0**;** i **<=** N**-**1**;** i**++)** **{**  **for(**j **=** 0**;** j **<=** N**-**1**;** j**++)**  scanf**(**"%d"**,** **&**a**[**i**][**j**]);**  **}**  printf**(**"Matrix A is: \n"**);**  **for(**i **=** 0**;** i **<=** N**-**1**;** i**++)** **{**  **for(**j **=** 0**;** j **<=** N**-**1**;** j**++)** **{**  printf**(**"%5d"**,** a**[**i**][**j**]);**  b**[**j**][**i**]** **=** a**[**i**][**j**];**  **}**  printf**(**"\n"**);**  **}**  printf**(**"Matrix A^T is:\n"**);**  **for(**j **=** 0**;**j **<=** N**-**1**;** j**++)** **{**  **for(**i **=** 0**;** i **<=** N**-**1**;** i**++)**  printf**(**"%5d"**,** b**[**j**][**i**]);**  printf**(**"\n"**);**  **}**  **return** 0**;**  **}** |

### 运行结果



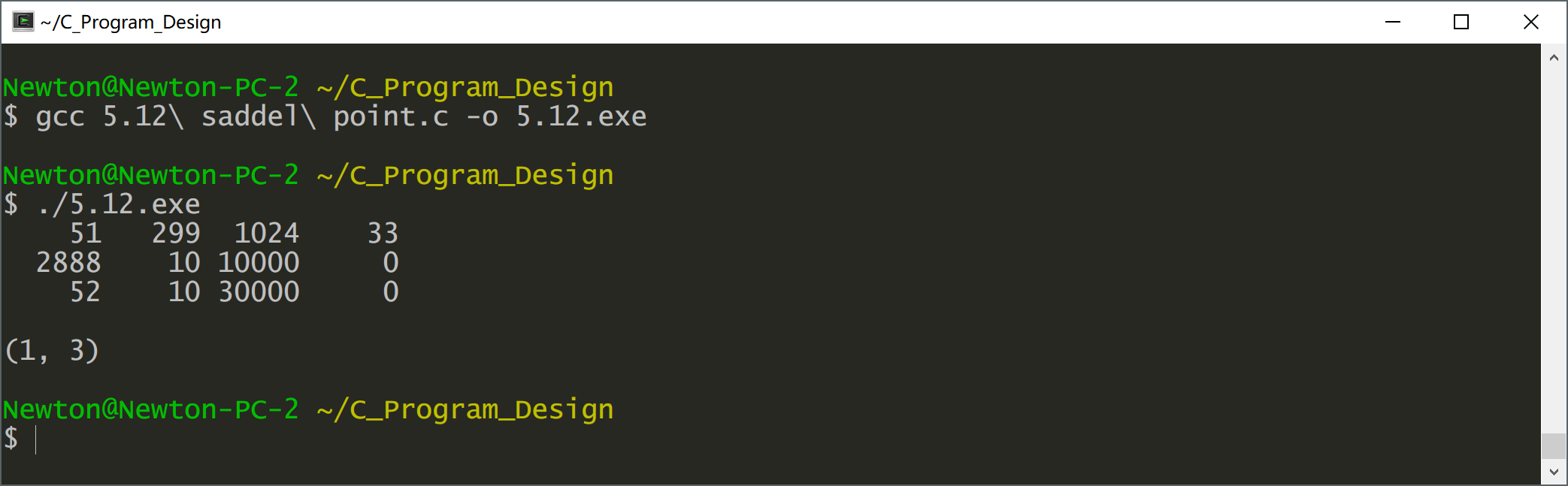
## 矩阵鞍点

从键盘输入行列式，并输出此行列式；然后求所有的鞍点（某元素若是本行元素中的最大者，同时又是本列元素中最小者，则此元素称为鞍点）。最后输出这些鞍点及其对应坐标值。（若无鞍点，则显示无鞍点信息）。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47 | /\*  \* filename: 5.12 saddel point.c  \* property: exercise  \*/  #include <stdio.h>  int main**()** **{**  int a**[**3**][**4**]** **=** **{**  **{**51**,** 299**,** 1024**,** 33**},**  **{**2888**,** 10**,** 10000**,** 0**},**  **{**52**,** 10**,** 30000**,** 0**}**  **};**  int i**,** j**,** k**;**  **for** **(**i **=** 0**;** i **<** 3**;** i**++)** **{**  **for** **(**j **=** 0**;** j **<** 4**;** j**++)** **{**  printf**(**"%6d"**,** a**[**i**][**j**]);**  **}**  printf**(**"\n"**);**  **}**  printf**(**"\n"**);**  int g1 **=** 1**;**  int g2 **=** 1**;**  **for** **(**i **=** 0**;** i **<** 3**;** i**++)** **{**  **for** **(**j **=** 0**;** j **<** 4**;** j**++)** **{**  **for** **(**k **=** 0**;** k **<** 3**;** k**++)** **{**  **if** **(**a**[**k**][**j**]** **<** a**[**i**][**j**])** **{**  g1 **=** 0**;**  **}**  **}**  **for** **(**k **=** 0**;** k **<** 4**;** k**++)** **{**  **if** **(**a**[**i**][**k**]** **>** a**[**i**][**j**])** **{**  g2 **=** 0**;**  **}**  **}**  **if(**g1 **\*** g2 **==** 1**)**  printf**(**"(%d, %d)\n"**,** i**+**1**,** j**+**1**);**  g1 **=** g2 **=** 1**;**  **}**  **}**  **return** 0**;**  **}** |

### 运行结果



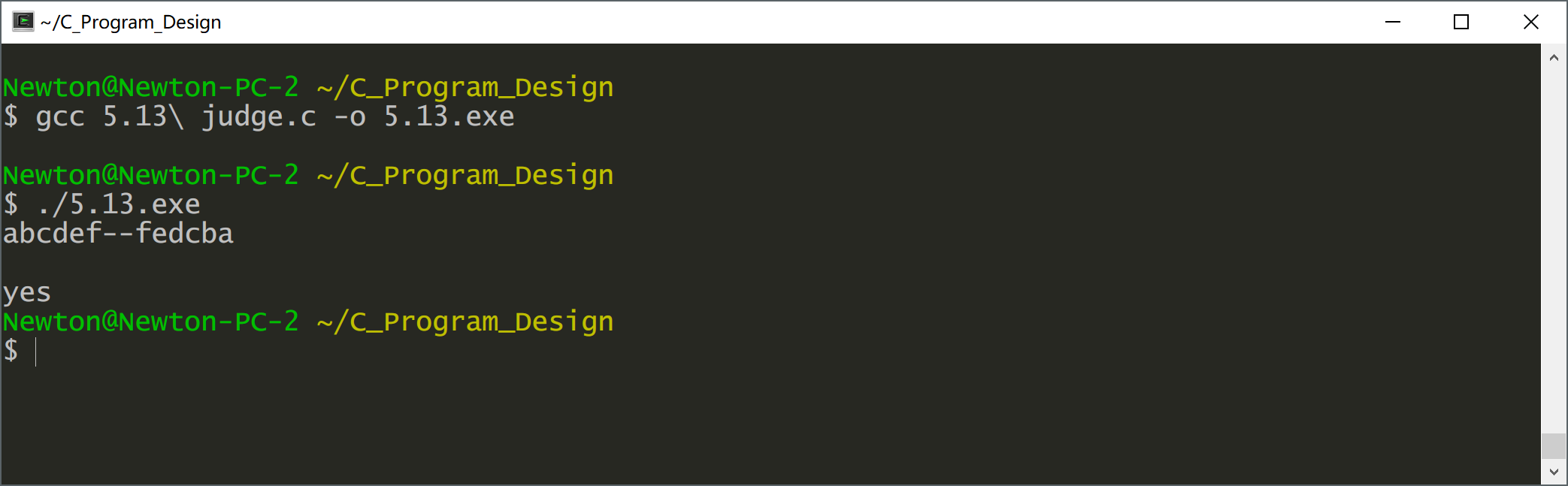
## 回文字符串判断

从键盘上输入一字符串，并判断是否形成回文（即正序和逆序一样，如“abcd dcba”）。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | /\*  \* filename: 5.13 judge.c  \* property: exercise  \*/  #include <stdio.h>  #include <string.h>  int main**()** **{**  char a**[**100**];**  gets**(**a**);**  **if** **(**strlen**(**a**)** **%** 2 **==** 1**)** **{**  printf**(**"not 1"**);**  **return** 0**;**  **}**  int n **=** strlen**(**a**)** **/** 2**;**  int i**,** j **=** 0**;**  char b**[**50**]** **=** ""**;**  puts**(**b**);**  **for** **(**i **=** 2**\***n **-** 1**;** i **>=** n**;** i**--)** **{**  b**[**j**++]** **=** a**[**i**];**  **}**  a**[**n**]** **=** '\0'**;**  **if** **(**strcmp**(**a**,** b**)** **==** 0**)** **{**  printf**(**"yes"**);**  **}**  **return** 0**;**  **}** |

### 运行结果



## 字符串修改

按字符数组输入字符，将其中的英文字母都改成‘\*’，然后按字符串形式输出。

### 程序代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | /\*  \* filename: 5.14 string.c  \* property: exercise  \*/  #include <stdio.h>  #include <string.h>  int main**()** **{**  char a**[**100**];**  gets**(**a**);**  int i**;**  **for** **(**i **=** 0**;** i **<** strlen**(**a**);** i**++)** **{**  **if** **(**a**[**i**]** **>** 'a' **&&** a**[**i**]** **<** 'z' **||** a**[**i**]** **>** 'A' **&&** a**[**i**]** **<** 'Z'**)** **{**  a**[**i**]** **=** '\*'**;**  **}**  **}**  puts**(**a**);**  **}** |

### 运行结果



# 实验总结

数组作为指针的前身，本身具有很多优越性，比如大小固定，而且申请释放不需要自己管理。大小固定既是一种有点又是一种缺点，在后来的指针程序设计中，很多语句来得不如用数组轻松，但是在编写灵活性强的结构时，数组就力所不及了。本实验中的数组关注string.h给出的几个标准函数。

本次实验，集中主要精力，在以前版本的基础上，对文档结构进行了重整，看起来自然了很多，目录也规范了很多。有关编程的规范性问题，参考林锐高质量C/C++编程指南的第一版[3]。

# 参考文献

1. Stevens, W.R. and S.A. Rago, *UNIX环境高级编程*. 2nd ed. 2005, 北京: 人民邮电出版社.

2. Hahn, H., *Harley Hahn's Guide to Unix and Linux*. 2009, New York: McGraw-Hill.

3. **林锐**, *高质量 C++/C 编程指南*. 1.0 ed. 2001.

# 教师评语