

# 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 13 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

### 一、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。

### 二、实验环境

Windows10 Pro Workstation 17096;  
Code::Blocks 16.01 GCC 集成开发环境;  
Cygwin GCC 编译器。

### 三、实验内容

#### 3.1 有关概念

1. 只有静态数组和外部数组才能初始化。
2. 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。
3. 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。
4. 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，在其中存放一个字符串 `"I love china"`。
5. 用数组名作函数实参时，传到形参的是数组的首地址。

#### 3.2 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。

### 3.2.1 程序代码

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

### 3.2.2 运行结果

```

~/C_Program_Design
Newton@Newton-PC-2 ~/C_Program_Design
$ gcc 5.1\ print\ test.c -o 5.1.exe

Newton@Newton-PC-2 ~/C_Program_Design
$ ./5.1.exe
  1   1   1   1   1
  2   2   2   2   2
 c   c   c   c   c
a[5] = 0
b[2][2] = c
I am a boy
1 1 1 1 1
0 0 0 2 2
2 2 1667457891 1667457891 25443
0 1835081801 1646289184 31087 0

Newton@Newton-PC-2 ~/C_Program_Design
$

```

### 3.3 排序

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10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;
```

#### 3.3.1 程序代码

```

1  #include<stdio.h>
2  int main()
3  {
4      int i,array[10],min,k=0;
5      printf("Please input 10 data\n");
6      for(i=0;i<10;i++)
7          scanf("%d",&array[i]);
8      printf("Before exchange:\n");
9      for(i=0;i<10;i++)
10         printf("%5d",array[i]);
11     min=array[0];
12     printf("\n");
13     for(i=1;i<10;i++)
14         if(min>array[i])
15         {
16             min=array[i];
17             k=i;
18         }
19     array[k]=array[0];
20     array[0]=min;
21     printf("After exchange:\n");
22     for(i=0;i<10;i++)
23         printf("%5d",array[i]);

```

```

24     printf("\n");
25     printf("k=%d\t min=%d\n",k,min);
26     return 0;
27 }

```

### 3.3.2 运行结果

## 3.4

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）。

### 3.4.1 程序代码

```

1  #include<stdio.h>
2  int main()
3  {
4      int i,n;
5      float a,x[20],y[21];
6      printf("Please input n value\n");
7      scanf("%d",&n);
8      printf("Please input value (from small to big)\n");
9      for(i=0;i<n;i++)
10         scanf("%f",&x[i]);
11     printf("Insert value=?");
12     scanf("%f",&a);
13     i=0;
14     while(a>x[i]&& i<n)
15     {
16         y[i]=x[i];
17         i++;
18     }
19     y[i]=a;
20     for(i=i+1;i<n+1;i++)
21         y[i]=x[i-1];
22     printf("\n");
23     for(i=0;i<n+1;i++)
24     {
25         printf("%8.2f",y[i]);
26         if((i+1)%5==0) puts("\n");
27     }
28     return 0;
29 }

```

### 3.4.2 运行结果

### 3.5

编写一程序，一班级有  $n$  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。

#### 3.5.1 程序代码

```

1  #include<stdio.h>
2  void strup(char str[])
3  int main()
4  {
5      char str[20];
6      char name[20][20];
7      int i,j,t,n;
8      printf("Please input name number of sorting\n");
9      scanf("%d",&n);
10     printf("Please input name\n");
11     for(i=0;i<n;i++)
12     {
13         gets(name[i]);
14         strup(name[i]);
15     }
16     for(i=0;i<n;i++)
17     {
18         for(j=i+1;j<n;j++)
19         {
20             for(k=0;;k++)
21             if(name[i][k]<name[j][k])
22                 break;
23             else
24                 if(name[i][k]>name[j][k])
25                 {
26                     strcpy(str,name[j]);
27                     strcpy(name[j],name[i]);
28                     strcpy(name[i],str);
29                     break;
30                 }
31         }
32     }
33     for(i=0;i<n;i++)
34     printf("%s",name[i]);
35 }
36 void strup(char str[])
37 {
38     int i;
39     for(i=0;str[i]!='\0';i++)
40     if(str[i]>='a' && str[i]<='z')
41         str[i]=str[i]+'A'-'a';
42 }
```

### 3.5.2 运行结果

### 3.6 \* 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

8	1	6
3	5	7
4	9	2

要求打印由1到 $n^2$ 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 $n \times n$ 止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 $n$ （指最下一行）；

当上一个数的列数为 $n$ 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 $n$ 列时，则把下一个数放在上一个数的下面。

```

1  #include<stdio.h>
2  int main()
3  {
4      int a[16][16],i,j,k,p,m,n;
5      p=1; /* 初始化 */
6      while(p==1)
7      {
8          printf("Please input n:\n");
9          scanf("%d",&n);
10         if (n!=0) && (n<=15) && (n%2!=0)
11         {
12             printf("矩阵阶数是%d\n",n);
13             p=0;
14         }
15     }
16     for(i=1;i<=n;i++)
17     for(j=1;j<=n;j++)
18         a[i][j]=0;
19     j=n/2+1; /* 建立魔方面 */
20     a[1][j]=1;
21     for(k=2;k<= n*n;k++)
22     {
23         i=i-1;
24         j=j+1;
25         if((i<1) && (j>n))
26         {
27             i=i+2;
28             j=j-1;
29         }

```

```

30     else
31     {
32         if(i<1) i=n;
33         if(j>n) j=1;
34     }
35     if(a[i][j]==0)
36         a[i][j]=k;
37     else
38     {
39         i=i+2;
40         j=j-1;
41         a[i][j]=k;
42     }
43 }
44 for(i=1;i<=n;i++)          /* 输出 */
45 {
46     for(j=1;j<=n;j++)
47     {
48         printf("%4d",a[i][j]);
49         printf("\n");
50     }
51 }
52 }

```

### 3.6.1 运行结果

## 3.7 元素移位

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，即按如下操作：  
首先将光标移至 `t=a[9]` 的语句行上，按 F4，再用 Ctrl + F7 操作将 `a` 数组的内容显示出来，然后将光标移至 `a[i]=a[i-1]` 的语句行上，不断按 F4，观察 `a` 数组值的变化情况，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？

1. 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。

### 3.7.1 程序代码

```

1  #include<stdio.h>
2  int main()
3  {
4      int i,t,a[10]={0,1,2,3,4,5,6,7,8,9};
5      t=a[9];
6      for(i=1;i<10;i++)

```

```

7     a[i]=a[i-1];
8     a[0]=t;
9     printf("\n");
10    for(i=0;i<10;i++)
11        printf("%d" ,a[i]);
12 }

```

### 3.8 成绩排序

输入  $n$  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？  
以下是用选择法实现的排序。

#### 3.8.1 程序代码

```

1  #include<stdio.h>
2  int main( )
3  {
4      int i,j,t,n,a[n];
5      printf("\n n=?");
6      scanf("%d",&n);
7      printf("input n numbers :\n");
8      for (i=0;i<n;i++)
9          scanf("%d",&a[i]);
10     for(i=0;i<n-1;i++)
11         for(j=i+1;j<n;j++)
12             if(a[i]<a[j])
13                 {t=a[i];a[i]=a[j];a[j]=t;}
14     printf("the sorted numbers:\n");
15     for(i=0;i<n;i++)
16         printf("%4d",a[i]);
17     return 0;
18 }

```

### 3.9 矩阵操作

1. 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。



②以矩阵的形式输出对换后的矩阵。

### 3.10

2. P141 7.6。

### 3.11

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。

### 3.12 程序排错

1. 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];
int a[3]={10,8,3,4}
float a[4];a[0]=15;a[4]=100;
int i,j,k[3]={3};
char ch[]={};
```

### 3.13 排序

从键盘输入 10 整数，分别用选择法，交换法，插入法对它们进行由小到大（或由大到小）排序，且输出排序后的结果。

利用上题程序的排序结果，再从键盘输入一个整数，然后从排序的数组查找是否有与刚输入的数组相等的元素，若有，是第几个？若找不到，则显示相应的信息。

下面是一个输出 10 阶杨辉三角形的程序：

程序代码

```
1  #include<stdio.h>
2  #define N 10
3  int main()
4  {
5      int a[N][N],i,j;
6      for(i=1;i<N;i++)      /*形成主对角线 and 第 1 列元素*/
7      {
8          a[i][1]=1;
9          a[i][i]=1;
10     }
11     for(i=3;i<N;i++)      /* 形成其余元素*/
12         for(j=2;j<=i-1;j++)
13             a[i][j]=a[i-1][j-1]+a[i-1][j];
14     for(i=1;i<N;i++)      /* 输出*/
15     {
16         for(j=1;j<30-2*i;j++)
17             printf(" ");
18         for(j=1;j<=i;j++)
19             printf("%-4d",a[i][j]);
20         printf("\n");
```

```

21     }
22     return 0;
23 }

```

修改上述程序，使输出形式为

```

      1 2
    1 2 3
  1 2 3 4

```

（一直到第 10 行）

### 3.14 矩阵转置

从键盘输入  $N \times N$  的矩阵（ $N$  可以定义为 5），输出此矩阵和转置后的矩阵。

### 3.15 矩阵鞍点

从键盘输入  $m \times n$  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）。

### 3.16 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”）。

### 3.17 字符串修改

5. 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。

## 四、实验过程

1 题：

这个题目看重两个方面，一是设计数组，二是以美观的形式输出数组。

```

1  #include<stdio.h>
2  #define N 10
3  int main()
4  {
5      int a[N][N],i,j;
6      for(i=1;i<N;i++)      /*形成第 1 与第 2 列元素*/
7      {
8          a[i][1]=1;
9          a[i][2]=2;
10     }
11     for(i=2;i<N;i++)      /* 形成其余元素*/
12         for(j=3;j<=i;j++)
13             a[i][j]=a[i-1][j-1]+1;
14     for(i=1;i<N;i++)      /* 输出*/

```

```

15     {
16         for(j=1;j<30-2*i;j++)
17             printf(" ");
18         for(j=1;j<=i;j++)
19             printf("%-4d",a[i][j]);
20         printf("\n");
21     }
22     return 0;
23 }

```

2 题:

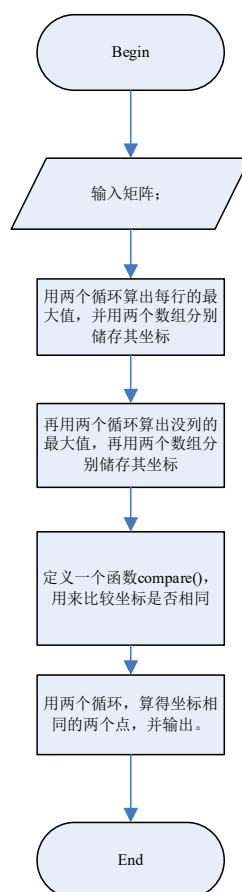
互换单个元素。

```

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

```

3 题:



## 矩阵鞍点

```

1  #include <stdio.h>
2  #define M 3 /* 行*/
3  #define N 4 /* 列*/
4  int main()
5  {
6      int compare(int a,int b,int c,int d);
7      printf("-----\n");
8      printf("Please input the elements of your matrix with ENTER to apart:\n");
9      int a[M][N],i,j;
10     int max_row[M],max_column[N];
11     int row,column;
12     int row_1[M],column_1[M]; /* 行的最大值有M个 */
13     int row_2[N],column_2[N]; /* 列的最大值有N个 */
14     for(i=0;i<=M-1;i++) /* 输入矩阵 */
15     {
16         for(j=0;j<=N-1;j++)
17             scanf("%d",&a[i][j]);
18     }
19     printf("The matrix is:\n");
20     for(i=0;i<=M-1;i++)
21     {
22         for(j=0;j<=N-1;j++)
23             printf("%5d",a[i][j]);
24         printf("\n");
25     }
26 }

```

```

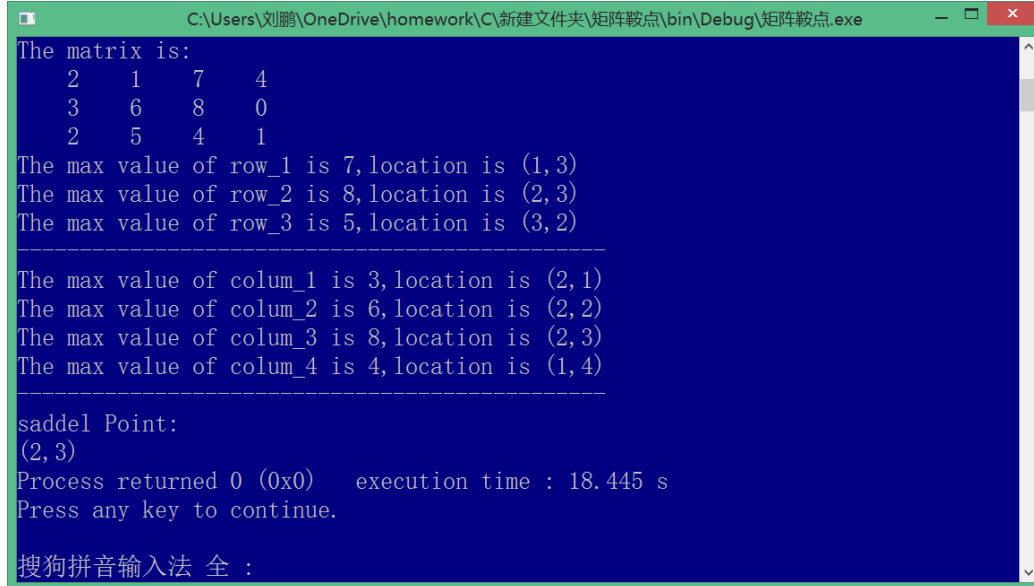
24     }
25     for(i=0;i<=M-1;i++)      /*求每行的最大值及其坐标*/
26     {
27         max_row[i]=a[i][0];
28         for(j=1;j<=N-1;j++)
29         {
30             if(a[i][j]>=max_row[i])
31             {
32                 max_row[i]=a[i][j];
33                 row=i;
34                 column=j;
35             }
36         }
37         row_1[i]=row+1;      /*得到了行最大的行坐标*/
38         column_1[i]=column+1; /*得到了行最大的列坐标*/
39         printf("The      max      value      of      row_%d      is      %d,location      is
40 (%d,%d)\n",row+1,max_row[i],row+1,column+1);
41     }
42     printf("-----\n");
43     for(j=0;j<=N-1;j++)      /*求每列的最大值及其坐标*/
44     {
45         max_column[j]=a[0][j];
46         for(i=0;i<=M-1;i++)
47         {
48             if(a[i][j]>=max_column[j])
49             {
50                 max_column[j]=a[i][j];
51                 row=i;
52                 column=j;
53             }
54         }
55         row_2[j]=row+1;
56         column_2[j]=column+1;
57         printf("The      max      value      of      column_%d      is      %d,location      is
58 (%d,%d)\n",column+1,max_column[j],row+1,column+1);
59     }
60     printf("-----\n");
61     printf("saddel Point:\n");
62     /*得到了 M+N 个数字，然后核对是否有相同位置的，是的话，就是鞍点*/
63     for(i=0;i<=M-1;i++)
64     {
65         for(j=0;j<=N-1;j++)
66         {
67             if(compare(row_1[i],column_1[i],row_2[j],column_2[j])==1)
68                 printf("(%d,%d)",row_1[i],column_1[i]);
69         }
70     }

```

```

71 int compare(int a,int b,int c,int d)
72 {
73     int z;
74     z=((a==c)&&(b==d))? 1:0;
75     return(z);
76 }
77

```



```

C:\Users\刘鹏\OneDrive\homework\C\新建文件夹\矩阵鞍点\bin\Debug\矩阵鞍点.exe
The matrix is:
 2  1  7  4
 3  6  8  0
 2  5  4  1
The max value of row_1 is 7,location is (1,3)
The max value of row_2 is 8,location is (2,3)
The max value of row_3 is 5,location is (3,2)
-----
The max value of colum_1 is 3,location is (2,1)
The max value of colum_2 is 6,location is (2,2)
The max value of colum_3 is 8,location is (2,3)
The max value of colum_4 is 4,location is (1,4)
-----
saddel Point:
(2,3)
Process returned 0 (0x0)   execution time : 18.445 s
Press any key to continue.
搜狗拼音输入法 全 :

```

很明显，鞍点就是 (2,3) , 8.

## 五、实验总结

随着时间的推移，我的编译器选择历经轮转，从最初的古老的 TC2.0，到 Code::Blocks 集成开发环境，再到 Visual Studio 2017，最终还是回到了 GNU 平台上来，使用开源的一套库进行实验。在此期间，云南大学也从一个普通的 211 大学跻身双一流大学行列，高级语言程序设计这门课程是否也该升级一下？TC2.0 这个编译器，界面十分古朴，在几十年前绝对算是一流的软件，但是现在，确实落后了，这主要是因为它无法编译在 64 位系统下运行的程序。但是 TC2.0 有很多的优势，比如完全可视化的编译过程，不会生成很多附加文件，这一点 Visual Studio 就太过专业化。经过对《UNIX 环境高级编程》[1]这本书的学习，还有诸如 *Harley Hahn's Guide to Unix and Linux*[2]这本书的阅读，我觉得基于 Shell 的 UNIX 环境似乎是最适合新手学习的。

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]。

## 六、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.

## 七、教师评语