

内容回顾

第六章 指针

- 指针基本概念
- 指向数组元素的指针
- 指针形参
- 数组形参
- 指向二维数组一整行的指针
- 指针数组
- 函数指针
- 函数指针作形参：
- 函数指针数组
- 返回指针值的函数

第七章 结构和链表

- 结构类型和结构变量
- 结构数组
- 结构与函数
- 链表

习题讲解

- 119-11
- 119-13
- 119-16
- 148-9
- 148-10
- 148-12
- 148-13
- 148-14
- 148-15
- 148-16
- 148-17
- 17-1
- 17-2
- 17-3
- 17-4
- 17-5
- 17-6
- 148-19
- 1126-1
- 1126-2
- 1126-3

期中练习

- 一 基本知识题
- 二 程序模拟运行题
 - 2.1
 - 2.2
 - 2.3
 - 2.4
- 三 程序理解题
 - 3.1
 - 3.2
 - 3.3
- 四 程序填空题
 - 4.1
 - 4.2
- 五 算法编程题

内容回顾

第六章 指针

指针基本概念

1. 指针，即内存块的地址，指针类型说明了该内存块的大小以及内存块中数据的类型
2. 定义指针变量：类型 *标识符
3. 引用指针所指的变量：*；取变量地址：&

- 注意

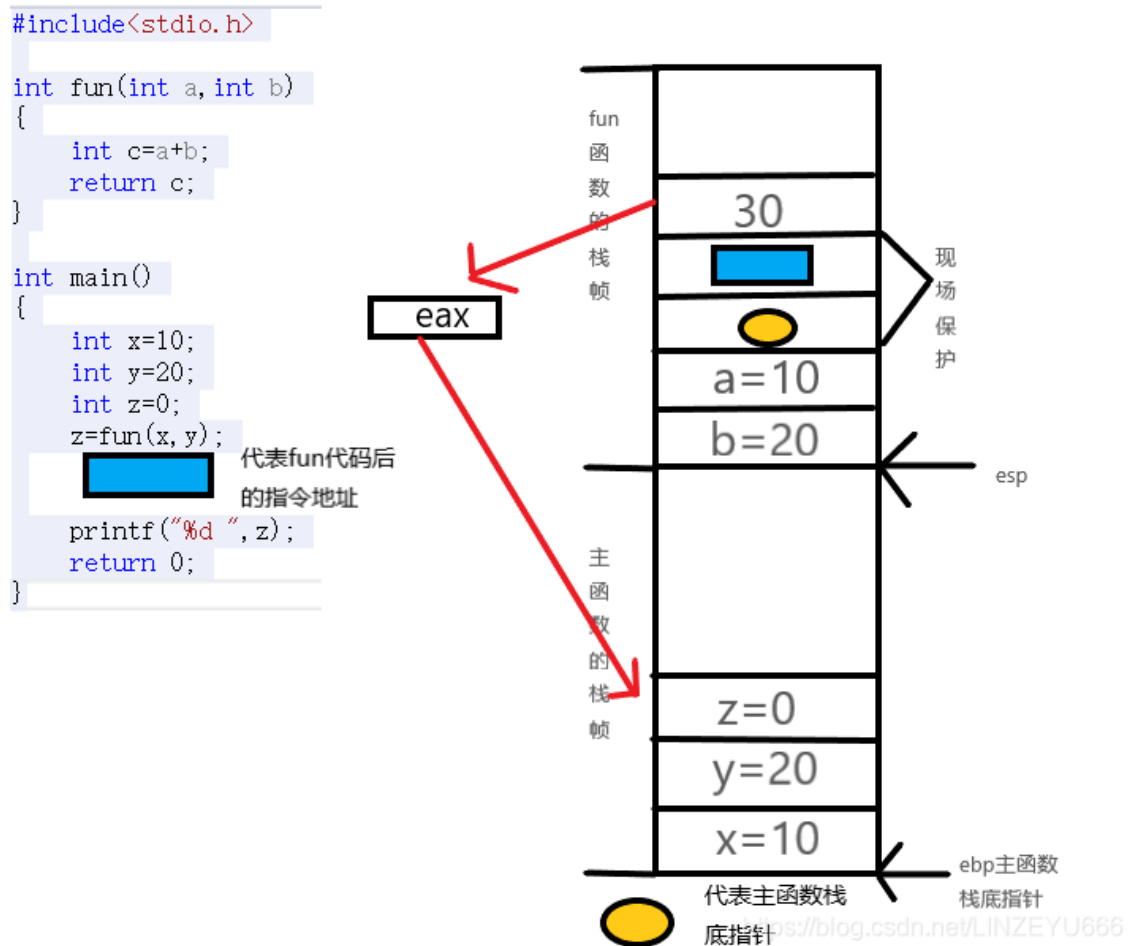
1. * 在定义指针和引用指针时的功能不同: `int *ip *ip = 1;`
2. * & ++ -- 从右向左结合: `x = (*ip)++; x = *ip++;`

指向数组元素的指针

- 一维数组名，即数组首元素的指针
- 对于 `int a[10]` ,
 - `x = a + 1;` 相当于 `x = (int*)((int)a + sizeof(*a));`
 - `y = x - a;` 相当于 `y = ((int)x - (int)a) / sizeof(*a);`
 - `*(a+1)` 相当于 `a[1]`
- 对于字符串指针
 - `char *s = "string";`
 - `char s[] = "string";`

指针形参

- C语言中的函数分为值传递或地址传递，使用指针形参可改变实参的值
- 函数的调用过程
 1. 建立栈帧空间
 2. 保护现场：主调函数运行状态和返回地址入栈
 3. 给形参分配存储空间，同时将主调函数的实参传递给形参；给被调函数的局部变量分配空间
 4. 执行被调函数
 5. 被调函数执行完成，释放其局部变量占用的栈空间
 6. 恢复现场：取主调函数运行状态及返回地址，释放栈帧空间
 7. 继续执行主调函数后续语句



- 为了在被调函数中改变实参的值，需要得到其地址，即指针形参，通过间接引用对所指变量进行修改

数组形参

- `void fun(int a[]);`
- `void fun(int* a);`
- `void fun(int a[][N]);`
- `void fun(int(*a)[N]);`

指向二维数组一整行的指针

```

1 int a[3][4];
2 int (*p)[4] = a;

```

指针数组

- `int *p[10];`
- 一般用来记录数据量较大的元素（结构体）的地址，通过调整指针数组中元素的位置，即改变记录顺序，可以实现更快速的排序功能

函数指针

- 函数指针，即函数的入口地址

```

1  int sum(int *, int);
2  int x = {1,2,3,4,5}, z;
3  // 定义方式1
4  typedef int(*sumPtType)(int *, int);
5  sumPtType sumPt = sum;
6  // 定义方式2
7  int (*fp)(int *, int) = sum;
8  // 调用
9  z = sum(x, 5);
10 z = (*sumPt)(x, 5);
11 z = (*fp)(x, 5);

```

- `*sumPtType` `*fp` 两侧的括号是必需的

函数指针作形参：

```

1  void excute(int (*fp)(int, int));
2  excute(fp);

```

函数指针数组

- `int (*fps[])(int, int) = {sum, min, max};`

返回指针值的函数

```

1  // 返回变量的指针的函数
2  int* fun(int, int);
3  // 返回函数指针的函数
4  int (*fun(int input_parameter1, int input_parameter2))(int, int);

```

第七章 结构和链表

结构类型和结构变量

```

1  struct Date
2  {
3      int day;
4      int month;
5      int year;
6  };
7  typedef struct
8  {
9      int day;
10     int month;
11     int year;
12 } DATE;
13
14 struct Date date1 = {16, 12, 2021}, date2;
15 //DATE date1, date2
16 date2.day = 16;
17 date2.month = 12;
18 date2.year = 2021;

```

- 结构变量占用的内存大小除了结构体内部成员变量的字节大小外，还受到字节对齐的影响
- 结构体内不能直接定义函数，但可以通过定义函数指针指向外部定义的函数来间接实现
- 引用结构成员
 1. 结构变量.成员名
 2. 结构指针->成员名
 3. (*结构指针).成员名

结构数组

- 定义
- 引用

结构与函数

- 结构形参
- 结构指针形参
- 返回结构的函数

链表

- 与占用连续内存位置的数组不同，链表一般针对不确定数量的元素，因此需要动态分配内存，各元素的内存位置是离散的，通过结构体内部的结构指针成员将各个元素连接起来，形成链表
- 链表特点
 1. 相较于数组，需要额外指针来连接各表元，占用的内存较多
 2. 各表元占用的内存可以不连续
 3. 相较于数组，插入、删除的时间复杂度较低；查询的时间复杂度较高
- 动态变量
 - `malloc()`, `calloc()` 用于在堆区申请动态空间，后者额外做清零操作
 - `free()` 只能作用于堆区申请的动态空间，且只能将申请到的空间一次性全部释放

```

1  #include<malloc.h>
2  int *arr1 = (int*)malloc(sizeof(int) * 100);
3  int *arr2 = (int*)calloc(100, sizeof(int));
4  free(arr1);
5  arr1 = NULL;
6  free(arr2);
7  arr2 = NULL;

```

- 单链表基本操作

```

1  typedef struct LinkedList
2  {
3      int val;
4      struct LinkedList *next;
5  } LList;
6  LList *head, *node, *pre, *cur;
7  int key;
8  int i;
9  // 建立空链表
10 head = pre = NULL;
11 for(i = 0; i < 3; i++)
12 {

```

```

13 // 创建表元
14 node = (LList*)malloc(sizeof(LList));
15 node->val = i;
16 node->next = NULL;
17 if (head == NULL)
18 {
19     head = pre = node;
20 }
21 else
22 {
23     pre->next = node;
24     pre = pre->next;
25 }
26 }
27 // 遍历
28 cur = head;
29 while (cur)
30 {
31     printf("%d\n", cur->val);
32     cur = cur->next;
33 }
34 // 查找
35 cur = head;
36 key = 1;
37 while (cur)
38 {
39     if (cur->val == key)
40     {
41         printf("%d existed\n", key);
42         break;
43     }
44     cur = cur->next;
45 }
46 if (!cur)
47 {
48     printf("No such key(%d)\n", key);
49 }

```

- 插入新表元
 - pre, node
 1. 首表元前
 2. 中间
 3. 尾表元后
- 删除表元
 - pre, node
 1. 首表元
 2. 中间
- 辅助表元：无需另外判断是否为首表元

习题讲解

119-11

1. auto
2. static

3. static
4. extern

119-13

```
1 #define JUDGE(x) ((!(x)%400) || ((x)%100 && !(x)%4)))
2 //非整百年份：能被4整除；整百年份：能被400整除
```

119-16

```
1 #ifndef PRINT1INT
2     #define PRINT1INT(x1) printf("%d\n", x1)
3 #endif
4 #ifndef PRINT2INT
5     #define PRINT2INT(x1, x2) printf("%d %d\n", x1, x2)
6 #endif
7 #ifndef PRINT3INT
8     #define PRINT3INT(x1, x2, x3) printf("%d %d %d\n", x1, x2, x3)
9 #endif
```

148-9

1. 定义长度为10的一维数组，类型为 `int`
2. 定义4*3的二维数组，类型为 `int`，`b[0][0]=1, b[3][0]=4, b[3][1]=2`，其余为0
3. 定义指针数组，元素类型为 `int*`
4. 定义指向二维数组一行的指针/定义指向长度为_的一维数组的指针，数组元素类型为 `int`
5. 定义函数指针，形参类型为 `(int, int)`，函数的返回值类型为 `int`
6. 定义返回指针的函数，形参类型为 `(int, int)`，返回值类型为 `int*`
7. 定义函数指针，形参类型为 `(int, int)`，函数的返回值类型为 `int*`

148-10

1. 函数1：输入所有学生的成绩表和指定学号，返回学号对应的学生成绩表
2. 函数2：输入某生的成绩表，打印成绩表内容，无返回值

```
1 #include<stdio.h>
2
3 int* search148_10(int (*scores)[5], int n, int key)
4 {
5     int i;
6     for (i = 0; i < n; i++)
7     {
8         if (**(scores + i) == key)
9         {
10             return *(scores + i);
11         }
12     }
13     return NULL;
14 }
15
16 void show148_10(int (*p)[5], int n)
17 {
18     int i;
```

```

19     char *description[5] = { "student id", "course 1", "course 2", "course
20     3", "course 4" };
21     if (p == NULL)
22     {
23         printf("No such student");
24         return;
25     }
26     for (i = 0; i < n; i++)
27     {
28         printf("%10s: %-3d\n", description[i], *(*p + i));
29     }
30     return;
31 }
32
33 int main()
34 {
35     int scores[3][5] = { 1,61,62,63,64,2,74,75,76,77,3,95,96,97,98 };
36     int key = 3;
37     int (*p)[5];
38     p = (int(*)[5])search148_10(scores, sizeof(scores)/sizeof(*p), key);
39     show148_10(p, sizeof(*p)/sizeof(int));
40 }

```

148-12

- 将原字符串划分为两部分，前半部分保存已处理的结果，后半部分用于遍历未处理字符。无论是否存在重复字符，前半部分和后半部分都不会重叠。
- 设置三个指针
 1. ch1用于遍历字符串中未处理的字符
 2. ch2用于遍历已保存的无重复字符
 3. ch3用于记录新的无重复字符

```

1  #include<stdio.h>
2
3  void removeRepeatedChar(char* s)
4  {

```



```

5      char *ch1 = s + 1, *ch2 = s, *ch3 = s + 1; // ch1遍历, ch2遍历去重, ch3记录
      去重
6      int flag;
7      if (*s == '\0') {
8          return;
9      }
10     while (*ch1)
11     {
12         flag = 0;
13         for (ch2 = s; ch2 < ch3; ch2++)
14         {
15             if (*ch1 == *ch2)
16             {
17                 flag = 1;
18                 break;
19             }
20         }
21         if (flag)
22         {
23             ch1++;
24             continue;
25         }
26         *ch3++ = *ch1++;
27     }
28     *ch3 = '\0';
29     return;
30 }
31
32 int main()
33 {
34     char s[100] = "aabbcccdabcd123ab345";
35     printf(" input: %s\n", s);
36     removeRepeatedChar(s);
37     printf("output: %s\n", s);
38
39 }

```

Microsoft Visual Studio 调试控制台

```

input: aabbcccdabcd123ab345
output: abcd12345
F:\Projects\C\Exercise\Debug\Exercise.exe (进程 58396) 已退出, 返回代码为: 0。
若要在调试停止时自动关闭控制台, 请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...

```

- 以函数指针为实参调用计算函数 excute()

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include<stdio.h>
3
4  int sum(int a, int b)
5  {
6      return a + b;
7  }
8  int diff(int a, int b)
9  {
10     return a - b;
11 }
12 int product(int a, int b)
13 {
14     return a * b;
15 }
16 int div(int a, int b)
17 {
18     return b == 0 ? 0 : a / b;
19 }
20 int excute(int (*fun)(int, int), int a, int b)
21 {
22     return (*fun)(a, b);
23 }
24
25 int main()
26 {
27     int a = 12, b = 3;
28     int (*funcs[])(int, int) = {sum, diff, product, div};
29     int (*fun)(int, int);
30     int selection;
31     char *menu[4] = { "+", "-", "*", "/" };
32
33     printf("#### MENU ####\n");
34     int i;
35     for (i = 0; i < 4; i++)
36     {
37         printf("[%d] %s\n", i+1, menu[i]);
38     }
39     printf("#####\n");
40     scanf("%d", &selection);
41     fun = funcs[selection-1];
42     printf("%d", excute(fun, a, b));
43     return 0;
44 }
```

```
Microsoft Visual Studio 调试控制台
#### MENU ####
[1] +
[2] -
[3] *
[4] /
#####
1
15
F:\Projects\C\Exercise\Debug\Exercise.exe (进程 74676)已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

148-14

```
1 char *strpos(char * s, char * t)
2 {
3     char *j, *k;
4     for (; *s; s++)
5     {
6         for (j = s, k = t; *k && *j == *k; j++, k++)
7         {
8             if (k != t && k == '\0')
9             {
10                return s;
11            }
12        }
13    }
14    return NULL;
15 }
```

- 判断字符串s中是否存在与字符串t相同的子串，若存在则返回s中以t开头的最大子串的首字符地址

148-15

```
1 #include<stdio.h>
2 int a = 2, c = 4;
3 f(int a, int *x)          //a=2, *x=4; a=40, *x=2
4 {
5     int b = 10;           //b=10;      b=10
6     static int c = 20;    //c=20;      c=35
7     b += a++;             //a=3, b=12; a=41, b=50
8     c += a + b;           //c=35;      c=126
9     *x = c + 2;           //*x=37;     *x=128
10 }
11 void main()
12 {
13     f(a, &c); //a=2, c=37
14     printf("In main(1): a=%d,c=%d\n", a, c);
15     f(3 + c, &a); //a=128, c=37
```

```
16     printf("In main(2): a=%d,c=%d\n", a, c);
17 }
```

In main(1): a=2,c=37

In main(2): a=128,c=37

148-16

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include<stdio.h>
3  #include<malloc.h>
4  #define N 10
5
6  void input(int* a[], int n)
7  {
8      int i;
9      for (i = 0; i < n; i++)
10     {
11         if (!scanf("%d", *(a + i)))
12         {
13             break;
14         }
15     }
16     for (; i < n; i++)
17     {
18         *(a + i) = 0;
19     }
20     return;
21 }
22 void output(int* a[], int n)
23 {
24     int i;
25     for (i = 0; i < n; i++)
26     {
27         printf("%4d", *(a + i));
28     }
29     printf("\n");
30     return;
31 }
32 void sort(int* a[], int n)
33 {
34     int i, j;
35     int* tmp;
36     for (i = 0; i < n - 1; i++)
37     {
38         for (j = n - 1; j > i; j--)
39         {
40             if (*(a + j) < *(a + j - 1))
41             {
42                 tmp = *(a + j);
43                 *(a + j) = *(a + j - 1);
44                 *(a + j - 1) = tmp;
45             }
46         }
47     }
48     return;
49 }
```

```

50
51 int main()
52 {
53     int *ap[N];
54     int i;
55     int n = N;
56     for (i = 0; i < N; i++)
57     {
58         ap[i] = (int*)malloc(sizeof(int));
59     }
60     input(ap, n);
61     printf(" input: ");
62     output(ap, n);
63     sort(ap, n);
64     printf("output: ");
65     output(ap, n);
66 }

```

```

Microsoft Visual Studio 调试控制台
1 3 2 4 6 8 7 5 0 9
input:  1  3  2  4  6  8  7  5  0  9
output: 0  1  2  3  4  5  6  7  8  9

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 66460) 已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...

```

148-17

17-1

```

1  #include<stdio.h>
2  #define N 3
3
4  void showMatrix(int arr[][N], int row, int col)
5  {
6      int i, j;
7      for (i = 0; i < row; i++)
8      {
9          for (j = 0; j < col; j++)
10         {
11             printf("%2d", (*(arr+i)+j));
12         }
13         printf("\n");
14     }
15     return;
16 }

```

```

17 void transpose(int arr[][N], int row, int col)
18 {
19     int i, j;
20     int tmp;
21     for (i = 0; i < row; i++)
22     {
23         for (j = 0; j < i; j++)
24         {
25             tmp = (*(arr + i) + j);
26             (*(arr + i) + j) = (*(arr + j) + i);
27             (*(arr + j) + i) = tmp;
28         }
29     }
30     return;
31 }
32
33 int main()
34 {
35     int arr[N][N] = { 1,2,3,4,5,6,7,8,9 };
36     printf("input: \n");
37     showMatrix(arr, N, N);
38     printf("\noutput: \n");
39     transpose(arr, N, N);
40     showMatrix(arr, N, N);
41 }

```

```

Microsoft Visual Studio 调试控制台
input:
1 2 3
4 5 6
7 8 9

output:
1 4 7
2 5 8
3 6 9

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 77612) 已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...

```

```

1  #include<stdio.h>
2
3  int main()
4  {
5      int length = 0;
6      char s[100] = "abcdefg";
7      char *ch = s;
8      while (*(ch++) != '\0')
9      {
10         length++;
11     }
12     printf("input: %s\nlength: %d", s, length);
13 }

```

Microsoft Visual Studio 调试控制台

```

input: abcdefg
length: 7
F:\Projects\C\Exercise\Debug\Exercise.exe (进程 44204) 已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...

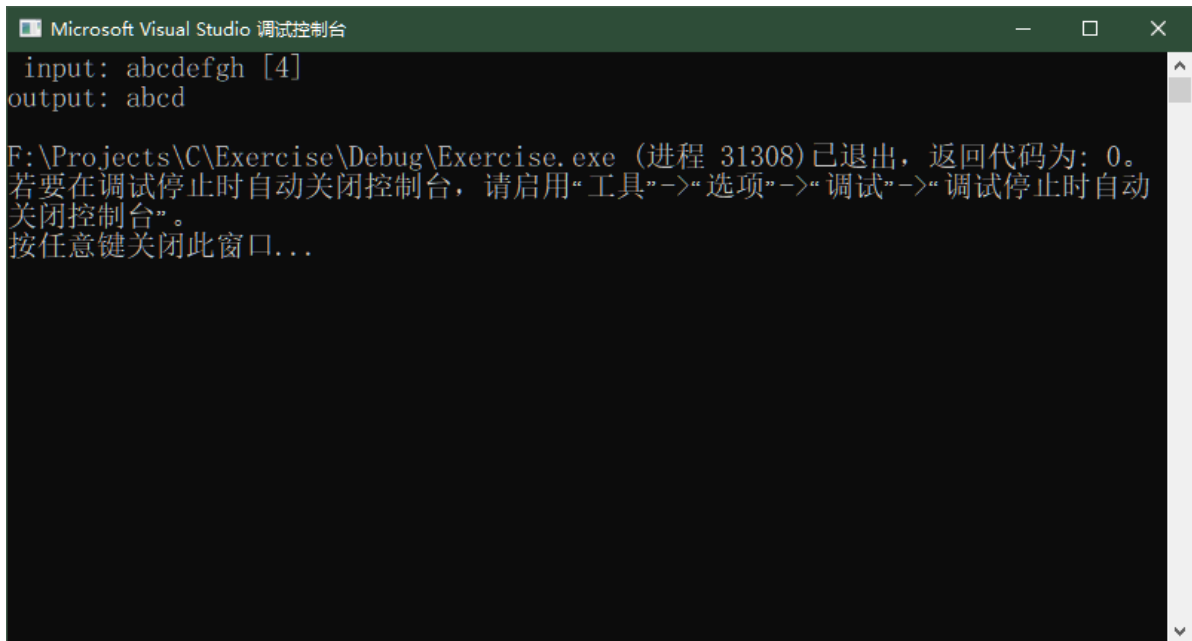
```

17-3

```

1  #include<stdio.h>
2
3  //int main148_17_3()
4  int main()
5  {
6      char s1[100], s2[100] = "abcdefgh";
7      int n = 4;
8      char *ch1 = s1, *ch2 = s2;
9
10     printf(" input: %s [%d]\n", s2, n);
11     while (n--)
12     {
13         *(ch1++) = *(ch2++);
14     }
15     *ch1 = '\0';
16     printf("output: %s\n", s1);
17 }

```



17-4

```
1  #include<stdio.h>
2
3  //int main148_17_4()
4  int main()
5  {
6      char s1[100] = "abcdefg", s2[100] = "1234567";
7      char *ch1 = s1, *ch2 = s2, ch;
8
9      printf("input s1: %s\n", s1);
10     printf("input s2: %s\n", s2);
11     while (*(ch1++) != '\0');
12     ch1--;
13     while ((ch = *(ch2++)) != '\0')
14     {
15         *(ch1++) = ch;
16     }
17     *ch1 = '\0';
18     printf("output: %s\n", s1);
19 }
```

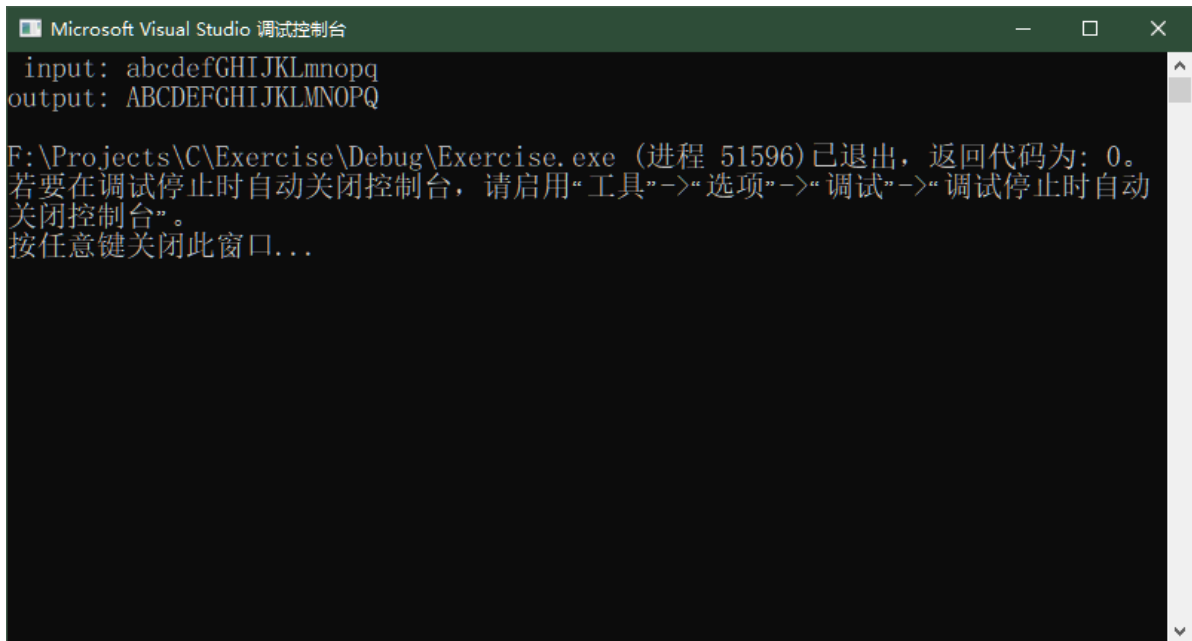


```
Microsoft Visual Studio 调试控制台
input s1: abcdefg
input s2: 1234567
output: abcdefg1234567

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 85304) 已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

17-5

```
1  #include<stdio.h>
2
3  //int main148_17_5()
4  int main()
5  {
6      char s[100] = "abcdefGHIJKLmnopq";
7      char *ch = s;
8      printf(" input: %s\n", s);
9      while (1)
10     {
11         if (!*ch)
12         {
13             break;
14         }
15         if (*ch >= 'a' && *ch <= 'z')
16         {
17             *ch += 'A' - 'a';
18         }
19         ch++;
20     }
21     printf("output: %s\n", s);
22 }
```



17-6

```
1  #include<stdio.h>
2
3  //int main148_17_6()
4  int main()
5  {
6      char s[100] = "    adfasdf    adf    ";
7      char *ch1 = s, *ch2 = s; //ch1保存, ch2遍历
8      int flag = 0; //0前一字符为空格, 1前一字符非空格
9
10     printf(" input: ~%s~\n", s);
11     while (1)
12     {
13         if (!*ch2)
14         {
15             break;
16         }
17         if (!flag && *ch2 == ' ')
18         {
19             ch2++;
20         }
21         else if (flag && *ch2 == ' ')
22         {
23             *ch1++ = *ch2++;
24             flag = 0;
25         }
26         else if (!flag && *ch2 != ' ')
27         {
28             flag = 1;
29             *(ch1++) = *(ch2++);
30         }
31         else if (flag && *ch2 != ' ')
32         {
33             *(ch1++) = *(ch2++);
34         }
35     }
36     if (*(ch1 - 1) == ' ')
```

```

37     {
38         *(ch1 - 1) = '\0';
39     }
40     else
41     {
42         *ch1 = '\0';
43     }
44     printf("output: ~%s~\n", s);
45     return 0;
46 }

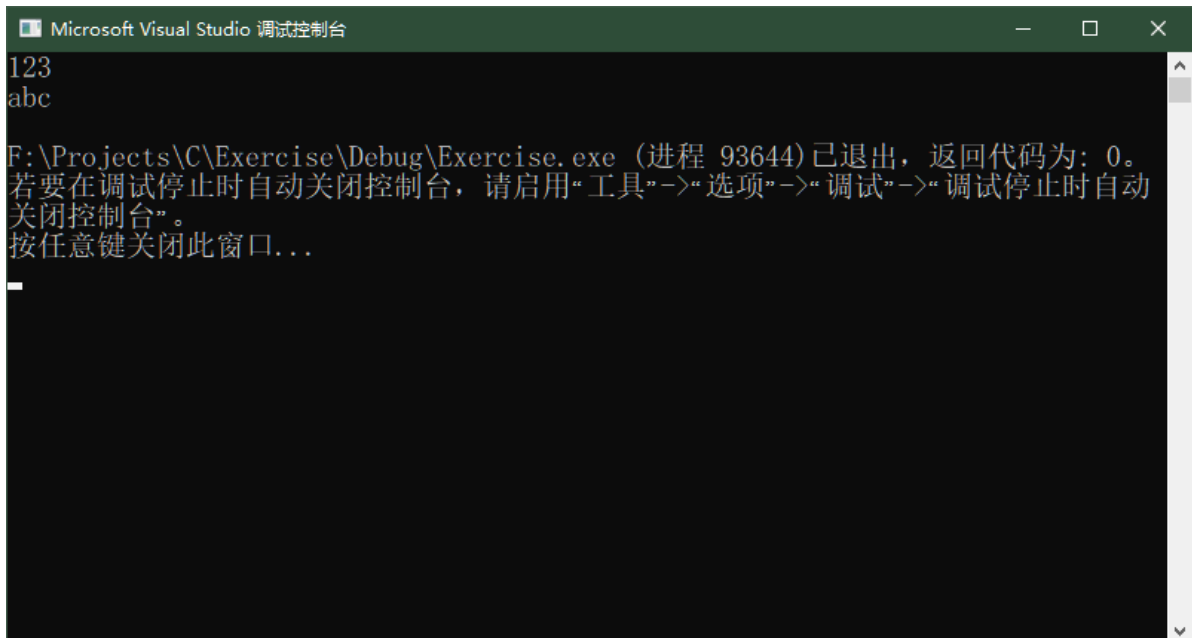
```

148-19

```

1 // 文件后缀须改为.cpp，引用为c++语法
2 #include<stdio.h>
3
4 void swap149_19(char *&s1, char *&s2)
5 {
6     char *s;
7     s = s1;
8     s1 = s2;
9     s2 = s;
10    return;
11 }
12
13 int main()
14 {
15     char *s1 = "abc", *s2 = "123";
16     swap149_19(s1, s2);
17     printf("%s\n%s\n", s1, s2);
18     return 0;
19 }

```



1126-1

输入10个整数，将其中最小的数与第1个数对换，把最大的数与最后一个数对换。写3个函数：1) 输入10个数；2) 进行处理；3) 输出10个数。

- 记录最大最小值及其索引，交换数值分为三步：
 - 交换最小值
 - 判断最大值索引是否为0，更新最大值索引
 - 交换最大值

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include<stdio.h>
3  #define N 10
4
5  void input1126_1(int *arr, int n)
6  {
7      int i;
8      for (i = 0; i < n; i++)
9      {
10         scanf("%d", &arr[i]);
11     }
12     return;
13 }
14
15 void process1126_1(int *arr, int n)
16 {
17     int i;
18     int val_max = arr[0], idx_max = 0, val_min = arr[0], idx_min = 0;
19     int tmp;
20     for (i = 1; i < n; i++)
21     {
22         if (arr[i] < val_min)
23         {
24             val_min = arr[i];
25             idx_min = i;
26         }
27         if (arr[i] > val_max)
28         {
```

```

29         val_max = arr[i];
30         idx_max = i;
31     }
32 }
33 arr[idx_min] = arr[0];
34 arr[0] = val_min;
35 idx_max = idx_max == 0 ? idx_min : idx_max;
36 arr[idx_max] = arr[n - 1];
37 arr[n - 1] = val_max;
38 return;
39 }
40
41 void output1126_1(int *arr, int n)
42 {
43     int i;
44     for (i = 0; i < n; i++)
45     {
46         printf("%-4d", arr[i]);
47     }
48     return;
49 }
50
51 //int main1126_1()
52 int main()
53 {
54     int arr[N];
55     input1126_1(arr, N);
56     process1126_1(arr, N);
57     printf("\n");
58     output1126_1(arr, N);
59     return 0;
60 }

```

1126-2

有n个整数，使前面各数顺序向后移m个位置，最后m个数变成最前面的m个数，写一函数实现上述功能，在主函数中输入n个整数和输出调整后的n个数。

- 参考p89 12

```

1  #include<stdio.h>
2  #define N 10
3
4  void reverse1126_2(int *a, int left, int right)
5  {
6      int tmp;
7      while (left < right)
8      {
9          tmp = a[left];
10         a[left] = a[right];
11         a[right] = tmp;
12         left++;
13         right--;
14     }
15     return;
16 }
17
18 void process1126_2(int *a, int n, int m)
19 {
20     reverse1126_2(a, 0, n - 1);
21     reverse1126_2(a, 0, n - m - 1);
22     reverse1126_2(a, n - m, n - 1);
23 }
24
25 int main()
26 {
27     int a[N] = { 1,2,3,4,5,6,7,8,9,10 };
28     int n = N, m = 3;
29     int i;
30     process1126_2(a, n, n-m);
31     for (i = 0; i < n; i++)
32     {
33         printf("%4d", a[i]);
34     }
35 }

```

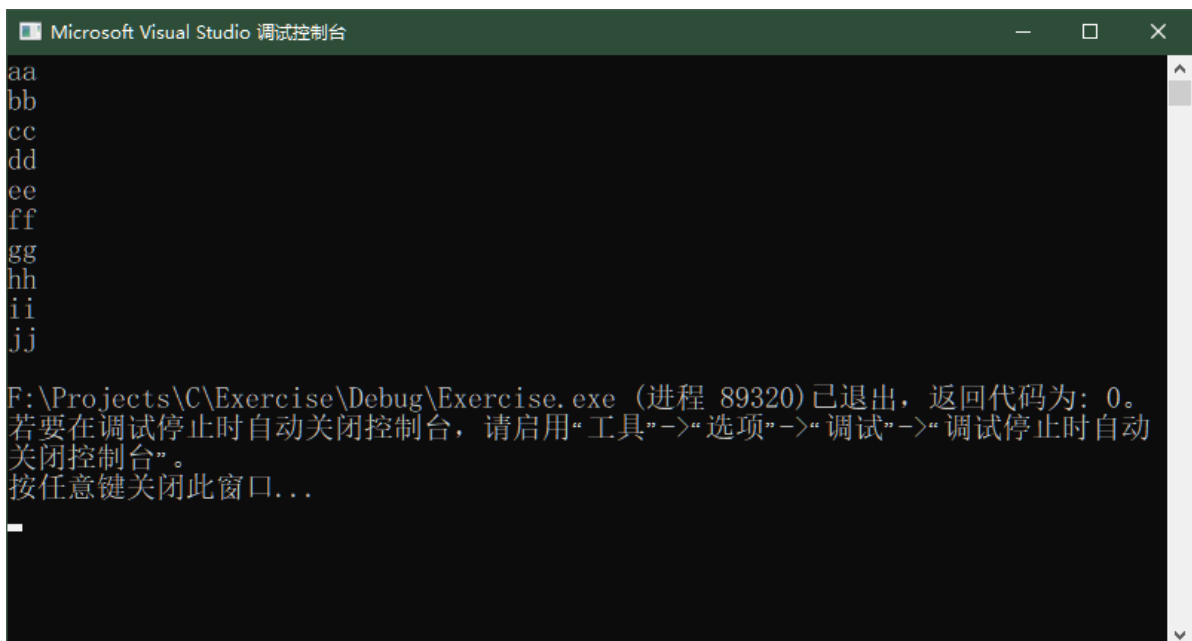
Microsoft Visual Studio 调试控制台

8 9 10 1 2 3 4 5 6 7

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 56596) 已退出，返回代码为: 0。
 若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
 按任意键关闭此窗口...

在主函数中输入10个字符串，用另一个函数对它们排序。然后在主函数输出排序好的结果。

```
1  #include<stdio.h>
2  #include<string.h>
3  #define N 10
4
5  void sort1126_3(char*s[], int n)
6  {
7      int i, j;
8      char *tmp;
9      for (i = 0; i < n - 1; i++)
10     {
11         for (j = n - 1; j > i; j--)
12         {
13             if (strcmp(s[i], s[j]) > 0)
14             {
15                 tmp = s[i];
16                 s[i] = s[j];
17                 s[j] = tmp;
18             }
19         }
20     }
21     return;
22 }
23
24 int main()
25 {
26     char *s[N] = { "aa", "ee", "ff", "gg", "hh", "bb", "cc", "dd", "ii",
27     "jj" };
28     int i;
29     sort(s, N);
30     for (i = 0; i < N; i++)
31     {
32         printf("%s\n", s[i]);
33     }
34 }
```



```
Microsoft Visual Studio 调试控制台
aa
bb
cc
dd
ee
ff
gg
hh
ii
jj
F:\Projects\C\Exercise\Debug\Exercise.exe (进程 89320) 已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动
关闭控制台”。
按任意键关闭此窗口...
```

期中练习

一 基本知识题

BBBBB DCC

二 程序模拟运行题

2.1

```
1 | 210
```

2.2

```
1 | 246
2 |
```

2.3

```
1 | 0
2 |
```

2.4

```
1 | 6
2 |
```

三 程序理解题

3.1

- 按顺序记录字符串中的数字，并转换为int型
- 123

3.2

```
1 | int search(char *word, char *tab[], int count[], int n)
2 | {
3 |     int low = 0, high = n - 1, mid; // 1
4 |     while (low <= high) // 2
5 |     {
6 |         mid = (low + high) / 2;
7 |         if (strcmp(word, tab[mid]) < 0) // 3
8 |         {
9 |             high = mid - 1;
10 |        }
11 |        else if (strcmp(word, tab[mid]) > 0) // 4
12 |        {
13 |            low = mid + 1;
14 |        }
15 |        else
16 |        {
17 |            return count[mid]; // 5

```



```

18     }
19 }
20 return -1;
21 }

```

3.3

```

1 int circle(int n, int d)
2 {
3     int s = 0;
4     int m;
5     m = n;
6     while(m) // 1
7     {
8         s = s * d + m % d; // 2
9         m /= d; // 3
10        //circle(m, d); // 4
11    }
12    return s == n;
13 }

```

四 程序填空题

4.1

- 4.1: `int *a, int n`
- 4.2: `a[i] == ret`
- 4.3: `ret = a[i], cnt = 1;`

```

1 #include<stdio.h>
2
3 int getMajorElement(int *a, int n)
4 {
5     int i, ret, cnt;
6     ret = a[0], cnt = 0;
7     for (i = 0; i < n; i++)
8     {
9         if (a[i] == ret)
10        {
11            cnt++;
12        }
13        else
14        {
15            cnt--;
16            if (cnt == 0)
17            {
18                ret = a[i], cnt = 1;
19            }
20        }
21    }
22    return ret;
23 }
24 int main()
25 {
26     int n, i;
27     int *a;

```

```

28     scanf_s("%d", &n);
29     a = (int*)malloc(n * sizeof(int));
30     for (i = 0; i < n; i++)
31     {
32         scanf_s("%d", &a[i]);
33     }
34     printf("%d\n", getMajorElement(a, n));
35     free(a);
36     return 0;
37 }

```

4.2

- 4.4: `value[next[p]] > x`
- 4.5: `next[i] = next[p];`
- 4.6: `value[next[p]]`

```

1  #include<stdio.h>
2  #define MAXN 1000
3
4  int value[MAXN] = { 0 };
5  int next[MAXN] = { 0 };
6
7  void insert(int x, int i)
8  {
9      int p = 0;
10     for (; next[p]; p = next[p])
11     {
12         if (value[next[p]] > x)
13         {
14             break;
15         }
16     }
17     value[i] = x;
18     next[i] = next[p];
19     next[p] = i;
20 }
21
22 int main()
23 {
24     int n, x;
25     int i;
26     int p;
27     scanf_s("%d", &n);
28     for (i = 1; i <= n; ++i)
29     {
30         scanf_s("%d", &x);
31         insert(x, i);
32     }
33     for (p = 0; next[p]; p = next[p])
34     {
35         printf("%d\n", value[next[p]]);
36     }
37     return 0;
38 }

```

五 算法编程题

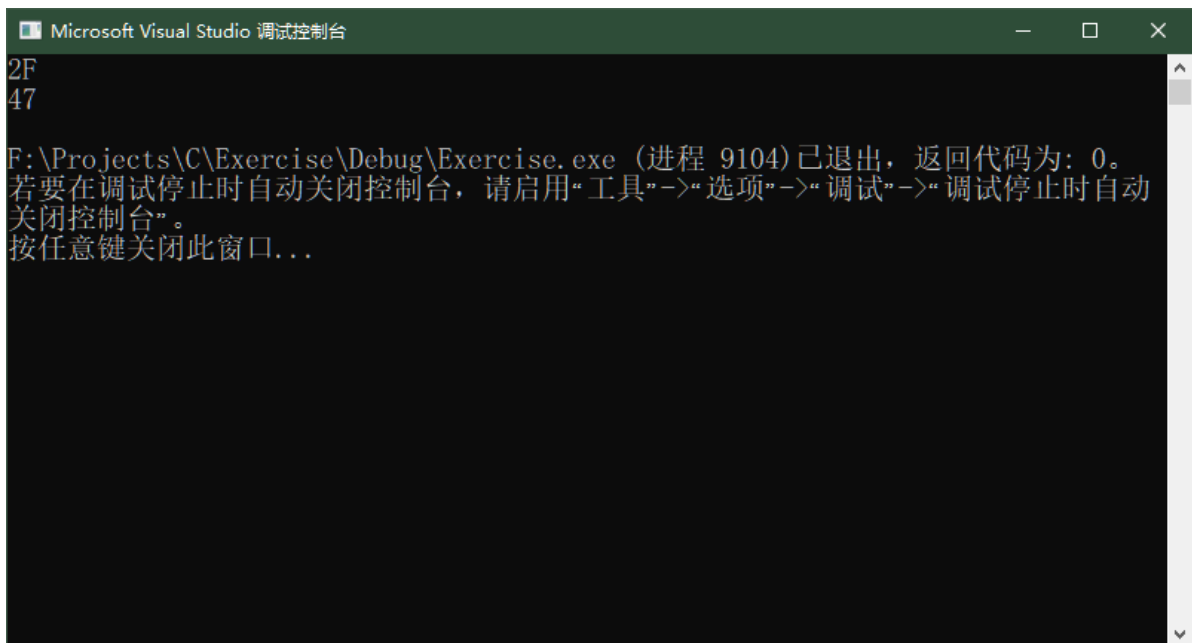
1

```
1  #include<stdio.h>
2  #define N 1<<10
3
4  static void input(char * str)
5  {
6      gets(str);
7  }
8
9  static int convert(char * str)
10 {
11     int res = 0, tmp;
12     while (*str)
13     {
14         if (*str >= '0' && *str <= '9')
15         {
16             tmp = *str - '0';
17         }
18         else if (*str >= 'A' && *str <= 'F')
19         {
20             tmp = *str - 'A' + 10;
21         }
22         res = res * 16 + tmp;
23         str++;
24     }
25     return res;
26 }
27
28 static int Maxprime(int n)
29 {
30     int i;
31     int flag;
32     if (n <= 1)
33     {
34         return 0;
35     }
36     while (n >= 2)
37     {
38         flag = 1;
39         for (i = 2; i*i <= n; i++)
40         {
41             if (n % i == 0)
42             {
43                 flag = 0;
44                 break;
45             }
46         }
47         if (flag)
48         {
49             return n;
50         }
51         n--;
52     }
53 }
```

```

54
55 static void output(int n)
56 {
57     printf("%d\n", n);
58 }
59
60
61 //int main1203_1()
62 int main()
63 {
64     char str[N];
65     int n, res;
66
67     input(str);
68     n = convert(str);
69     res = Maxprime(n);
70     output(res);
71 }

```



2

```

1  #include<stdio.h>
2
3  void comp(int n)
4  {
5
6      int x, i, num;
7      int sum;
8      int rec[100];
9      for (x = 2; x <= n; x++)
10     {
11         sum = 1;
12         num = 0;
13         rec[num++] = 1;
14         for (i = 2; i < x; i++)
15         {
16             if (x % i == 0)
17             {

```

```

18         sum += i;
19         rec[num++] = i;
20     }
21 }
22 if (sum == x)
23 {
24     printf("%d=1", x);
25     for (i = 1; i < num; i++)
26     {
27         printf("+%d", rec[i]);
28     }
29     printf("\n");
30 }
31 }
32 }
33
34 //int main1203_2()
35 int main()
36 {
37     comp(1000);
38 }

```

Microsoft Visual Studio 调试控制台

```

6=1+2+3
28=1+2+4+7+14
496=1+2+4+8+16+31+62+124+248

```

F:\Projects\C\Exercise\Debug\Exercise.exe (进程 6824) 已退出，返回代码为: 0。
 若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
 按任意键关闭此窗口...