

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级：**

**学 号：**

**姓 名：**

**指导教师：**

**报告日期：**

**软件学院**

**目 录**

[实验6 指针实验 3](#_Toc89887096)

[6.1、实验目的 3](#_Toc89887097)

[6.2、实验题目及要求 3](#_Toc89887098)

[6.2.1 源程序改错题 3](#_Toc89887099)

[6.2.2源程序完善、修改替换题 5](#_Toc89887100)

[3、跟踪调试题 14](#_Toc89887101)

[4、编程设计题 16](#_Toc89887102)

[6.3 实验小结 25](#_Toc89887103)

# 实验6 指针实验

6.1、实验目的

（1）熟练掌握指针的说明、赋值、使用。

（2）掌握用指针引用数组的元素，熟悉指向数组的指针的使用。

（3）熟练掌握字符数组与字符串的使用，掌握指针数组及字符指针数组的用法。

（4）掌握指针函数与函数指针的用法。

（5）掌握带有参数的main函数的用法。

6.2、实验题目及要求

6.2.1 源程序改错题

在下面所给的源程序中，函数strcopy(t, s)的功能是将字符串s复制给字符串t，并且返回串t的首地址。请单步跟踪程序，根据程序运行时出现的现象或观察到的字符串的值，分析并排除源程序的逻辑错误，使之能按照要求输出如下结果：

Input a string:

programming↙ （键盘输入）

programming

Input a string again:

language↙ （键盘输入）

language

#include<stdio.h>

char \*strcopy(char \*, const char \*);

int main(void)

{

char \*s1, \*s2, \*s3;

printf("Input a string:\n", s2);

scanf("%s", s2);

strcopy(s1, s2);

printf("%s\n", s1);

printf("Input a string again:\n", s2);

scanf("%s", s2);

s3 = strcopy(s1, s2);

printf("%s\n", s3);

return 0;

}

/\*将字符串s复制给字符串t，并且返回串t的首地址\*/

char \* strcopy(char \*t, const char \*s)

{

while(\*t++ = \*s++);

return (t);

}

**解答：**

（1）错误修改：

1) 第5行的字符数组定义错误，正确形式为：

#define maxn 10005

char s1[maxn] , s2[maxn] , s3[maxn] ;

2) 第12行的语句错误，正确形式为：

strcopy(s3, s2);

（2）错误修改后运行结果：

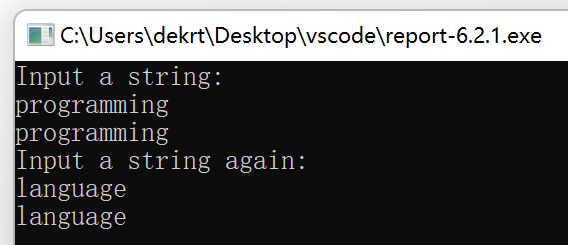


图6-1 源程序改错题的程序运行结果示意图

6.2.2源程序完善、修改替换题

（1）下面程序中函数strsort用于对字符串进行升序排序，在主函数中输入N个字符串（字符串长度不超过49）存入通过malloc动态分配的存储空间，然后调用strsort对这N个串按字典序升序排序。

①请在源程序中的下划线处填写合适的代码来完善该程序。

#include<stdio.h>

#include<\_\_\_\_\_\_\_\_\_>

#include<string.h>

#define N 4

/\*对指针数组s指向的size个字符串进行升序排序\*/

void strsort(char \*s[], int size)

{

\_\_\_\_\_\_\_temp;

int i, j;

for(i=0; i<size-1; i++)

for (j=0; j<size-i-1; j++)

if (\_\_\_\_\_\_\_\_\_\_\_)

{

temp = s[j];

\_\_\_\_\_\_\_\_\_\_\_;

s[j+1] = temp;

}

}

int main()

{

int i;

char \*s[N], t[50];

for (i=0; i<N; i++)

{

gets(t);

s[i] = (char \*)malloc(strlen(t)+1);

strcpy(\_\_\_\_\_\_\_);

}

strsort(\_\_\_\_\_\_\_\_);

for (i=0; i<N; i++) {puts(s[i]); free(s[i]);}

return 0;

}

**解答：**替换后的程序如下所示：

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#define N 4

/\*对指针数组s指向的size个字符串进行升序排序\*/

void strsort(char \*s[], int size)

{

char \*temp ;

int i, j;

for(i=0; i<size-1; i++)

for (j=0; j<size-i-1; j++)

if ( strcmp( s[j] , s[j+1]) > 0 )

{

temp = s[j];

s[j] = s[j+1] ;

s[j+1] = temp;

}

}

int main()

{

int i;

char \*s[N], t[50];

for ( i=0 ; i<N ; i++)

{

gets(t);

s[i] = (char \*)malloc(strlen(t)+1);

strcpy( s[i] , t );

}

strsort( s , N );

for (i=0; i<N; i++) {puts(s[i]); free(s[i]);}

return 0;

}

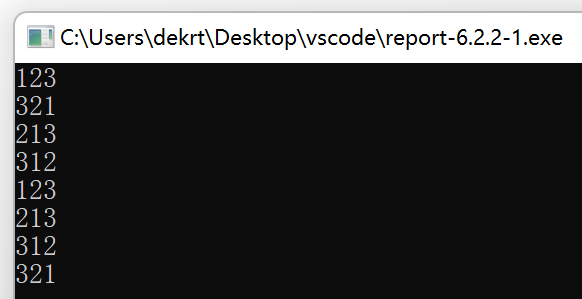


图6-2 源程序替换题的程序运行结果示意图

②数组作为函数参数其本质类型是指针。例如，对于形参char \*s[]，编译器将其解释为char \*\*s，两种写法完全等价。请用二级指针形参重写strsort函数，并且在该函数体的任何位置都不允许使用下标引用。

**解答：**替换后的程序如下所示：

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#define N 4

/\*对指针数组s指向的size个字符串进行升序排序\*/

void strsort(char \*\*s , int size)

{

char \*temp ;

int i, j;

for(i=0; i<size-1; i++)

for (j=0; j<size-i-1; j++)

if ( strcmp( \*( s + j ) , \*( s + j + 1 ) ) > 0 )

{

temp = \*( s + j ) ;

( \*( s + j ) ) = \*( s + j + 1 );

\*( s + j + 1 ) = temp;

}

}

int main()

{

int i;

char \*s[N], t[50];

for ( i=0 ; i<N ; i++)

{

gets(t);

s[i] = (char \*)malloc(strlen(t)+1);

strcpy( s[i] , t );

}

strsort( s , N );

for (i=0; i<N; i++) {puts(s[i]); free(s[i]);}

return 0;

}

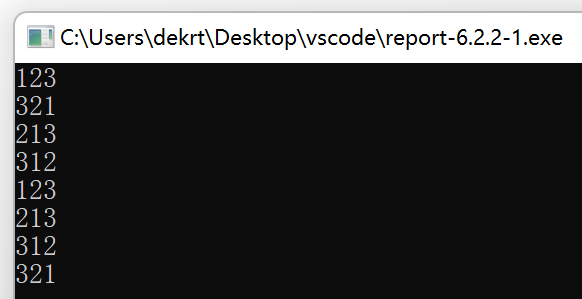


图6-3 源程序替换题的程序运行结果示意图

（2）下面源程序通过函数指针和菜单选择来调用库函数实现字符串操作；串复制strcpy、串连接strcat或串分解strtok。

①请在源程序中的下划线处填写合适的代码来完善该程序，使之能按照要求输出下面结果：

1 copy string.

2 connect string.

3 parse string.

4 exit.

input a number (1-4) please!

2↙ （键盘输入）

input the first string please!

the more you learn,↙ （键盘输入）

input the second string please!

the more you get. ↙ （键盘输入）

the result is the more you learn, the more you get.

# include<stdio.h>

# include<string.h>

int main (void)

{

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

char a[80], b[80], \*result;

int choice;

while(1)

{

do

{

printf("\t\t1 copy string.\n");

printf("\t\t2 connect string.\n");

printf("\t\t3 parse string.\n");

printf("\t\t4 exit.\n");

printf("\t\tinput a number (1-4) please.\n");

scanf("%d", &choice);

}while(choice<1 || choice>4);

switch(choice)

{

case 1: p = strcpy; break;

case 2: p = strcat; break;

case 3: p = strtok; break;

case 4: goto down;

}

getchar();

printf("input the first string please!\n");

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

printf("input the second string please!\n");

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

result = \_\_\_\_\_\_\_\_\_\_\_(a, b);

printf("the result is %s\n", result);

}

down:

return 0;

}

**解答：**替换后的程序如下所示：

# include<stdio.h>

# include<string.h>

int main (void)

{

char\* (\*p)( char a[] , const char b[] );

char a[80], b[80], \*result;

int choice;

while(1)

{

do

{

printf("\t\t1 copy string.\n");

printf("\t\t2 connect string.\n");

printf("\t\t3 parse string.\n");

printf("\t\t4 exit.\n");

printf("\t\tinput a number (1-4) please.\n");

scanf("%d", &choice);

}while(choice<1 || choice>4);

switch(choice)

{

case 1: p = strcpy; break;

case 2: p = strcat; break;

case 3: p = strtok; break;

case 4: goto down;

}

getchar();

printf("input the first string please!\n");

scanf("%s",a);

printf("input the second string please!\n");

scanf("%s",b);

result = p(a, b);

printf("the result is %s\n", result);

}

down:

return 0;

}

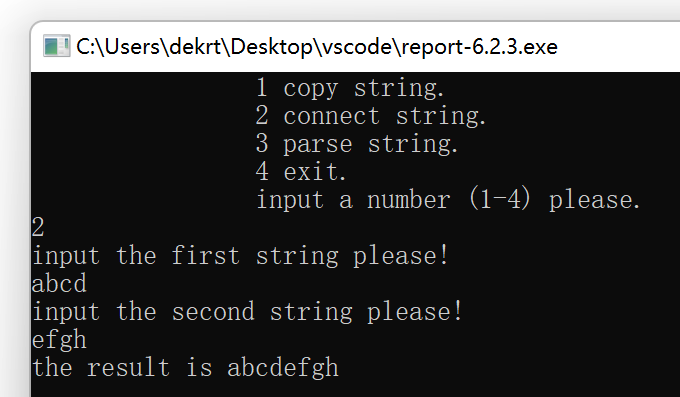


图6-4 源程序修改替换题的程序运行结果示意图

②函数指针的一个用途是用户散转程序，即通过一个转移表（函数指针数组）来实现多分枝函数处理，从而省去了大量的if语句或者switch语句。转移表中存放了各个函数的入口地址（函数名），根据条件的设定来查表选择执行相应的函数。请使用转移表而不是switch语句重写以上程序。

**解答：**替换后的程序如下所示：

# include<stdio.h>

# include<string.h>

char\* (\*ope[])( char a[] , const char b[] ) = { strcpy , strcat , strtok } ;

int main (void)

{

char\* (\*p)( char a[] , const char b[] );

char a[80], b[80], \*result;

int choice;

while(1)

{

do

{

printf("\t\t1 copy string.\n");

printf("\t\t2 connect string.\n");

printf("\t\t3 parse string.\n");

printf("\t\t4 exit.\n");

printf("\t\tinput a number (1-4) please.\n");

scanf("%d", &choice);

}while(choice<1 || choice>4);

if ( choice == 4 ) goto down;

getchar();

printf("input the first string please!\n");

scanf("%s",a);

printf("input the second string please!\n");

scanf("%s",b);

result = (ope[choice-1]) (a, b);

printf("the result is %s\n", result);

}

down:

return 0;

}

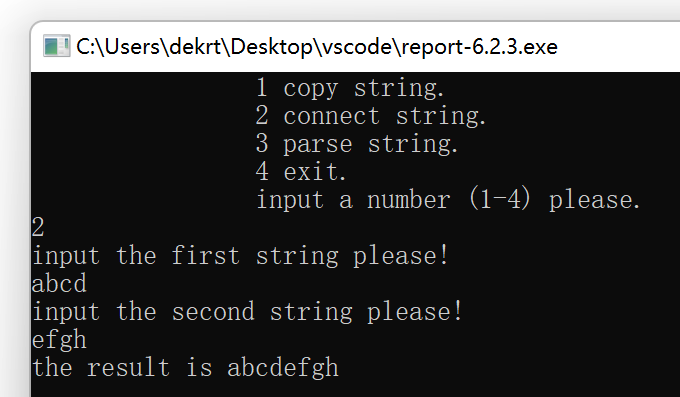


图6-5 源程序替换题的程序运行结果示意图

3、跟踪调试题

请按下面的要求对源程序进行操作，并回答问题和排除错误。

（1）单步执行。进入strcpy时watch窗口中s为何值？返回main时, watch窗口中s为何值？

（2）排除错误，使程序输出结果为：there is a boat on the lake.

#include "stdio.h"

char \*strcpy(char \*,char \*);

void main(void)

{

char a[20],b[60]="there is a boat on the lake.";

printf("%s\n",strcpy(a,b));

}

char \*strcpy(char \*s,char \*t)

{

while(\*s++=\*t++)

;

return (s);

}

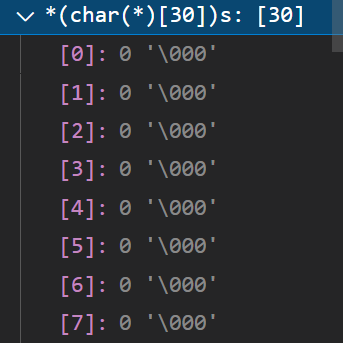


图6-6 跟踪调试题的程序运行结果示意图

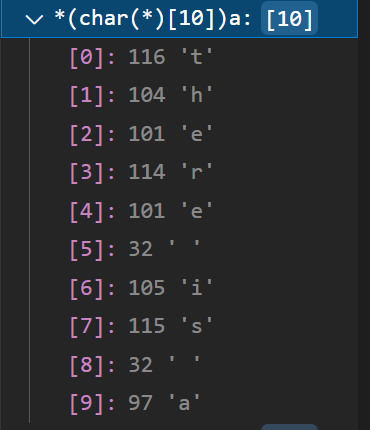


图6-7 跟踪调试题的程序运行结果示意图

替换后的程序如下所示：

#include "stdio.h"

char \*strcpy(char \*,char \*);

int main(void)

{

char a[20],b[60]="there is a boat on the lake.";

strcpy(a,b);

printf("%s\n",a);

return 0 ;

}

char \*strcpy(char \*s,char \*t)

{

while(\*s++=\*t++);

return (s);

}

4、编程设计题

（1）指定main函数的参数

在IDE（比如DevC++）中，选择“**运行**”｜“**参数**”菜单，在 “**传递给主程序的参数**”文本框中输入main函数的参数arg1 arg2 arg3，只输入命令行中文件名后的参数，文件名不作为参数输入，参数间以空格隔开。编写程序在命令行输出这三个参数。（注意不同IDE输入参数的方式不相同，可参考各个IDE的使用手册。）

1) 解题思路：

1.输入n个字符串

2.输出n个字符串

3.结束

2）程序清单

#include <stdio.h>

#define re register

int main( int argc , char\* argv[] )

{

for( re int i = 0 ; i < argc ; i++ )

{

printf("%s\n",argv[i]) ;

}

return 0 ;

}



图6-8 程序设计题1的程序运行结果示意图

以下（2）至（5）题对应Educoder 教学平台“C语言实验”课程，实验6，第13关实验6-1、第14关实验6-2、第15关实验6-3, 以及第16关实验6-4。

（2）一个长整型变量占4个字节，其中每个字节又分成高4位和低4位。输入一个长整型变量，要求从高字节开始，依次取出每个字节的高4位和低4位并以十六进制数字字符的形式进行显示，通过指针取出每字节。

样例输入：15

样例输出：0000000F

1) 解题思路：

1.输入k

2.定义指针p，类型为unsigned short

3.依次取出对应位置上的数字并输出

4.结束

2）程序清单

#include <stdio.h>

#define re register

unsigned long int k ;

int main()

{

scanf ("%u",&k) ;

unsigned short \*p = (unsigned short\*) &k ;

printf("%.4X%.4X",\*(p+1),\*p) ;

return 0 ;

}

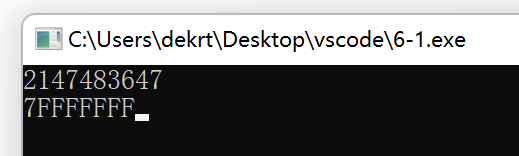


图6-9 程序设计题2的程序运行结果示意图

（3）旋转是图像处理的基本操作，编程实现一个将一个图像逆时针旋转90°。提示：计算机中的图像可以用一个矩阵来表示，旋转一个图像就是旋转对应的矩阵。将旋转矩阵的功能定义成函数，通过使用指向数组元素的指针作为参数使该函数能处理任意大小的矩阵。要求在main函数中输入图像矩阵的行数n和列数m，接下来的n行每行输入m个整数，表示输入的图像。输出原始矩阵逆时针旋转90°后的矩阵。

样例输入：

2 3

1 5 3

3 2 4

样例输出：

3 4

5 2

1 3

1) 解题思路：

1.输入n,m

2.输入对应的矩阵

3.对矩阵进行旋转并输出

4.结束

2）程序清单

#include <stdio.h>

#define re register

#define maxn 10

int n , m ;

int a[maxn][maxn] ;

int main()

{

scanf("%d%d",&n,&m) ;

for( re int i = 1 ; i <= n ;i++)

for( re int j = 1 ; j <= m ; j++)

scanf("%d",&a[i][j]) ;

for ( re int i = m ; i >= 1 ; i-- )

{

for( re int j = 1 ; j <= n-1 ; j++ )

printf("%d ",a[j][i]) ;

printf("%d\n",a[n][i]) ;

}

return 0 ;

}

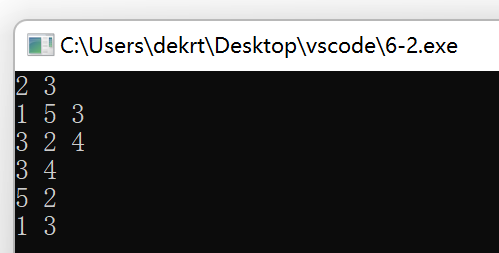


图6-10 程序设计题3的程序运行结果示意图

（4）输入n行文本，每行不超过80个字符，用字符指针数组指向键盘输入的n行文本，且n行文本的存储无冗余，删除每一行中的前置空格（' '）和水平制表符（'\t'）。要求：将删除一行文本中前置空格和水平制表符的功能定义成函数，在main函数中输出删除前置空格符的各行。

1) 解题思路：

1.读入字符串s

2.调用del()函数进行删除

3.在主函数中对字符串进行输出

4.结束

2）程序清单

#include <stdio.h>

#include <string.h>

#define re register

#define maxn 100

char s[maxn] ;

int cnt ;

void del( char s[] )

{

cnt = 0 ;

char tmp[maxn] ;

re int len = strlen(s) , i = 0 , j , k = 0 ;

while ( s[i++] == ' ' ) ;

i-- ;

for ( re int j = i ; j < len ; j++ )

if ( s[j] != '\t' )

tmp[k++] = s[j] ;

cnt = strlen(tmp) ;

for( re int i = 0 ; i < cnt ; i++ )

s[i] = tmp[i] ;

}

int main()

{

while ( gets(s) )

{

del(s) ;

for ( re int i = 0 ; i < cnt ; i++ )

printf("%c",s[i]) ;

putchar('\n') ;

}

return 0 ;

}

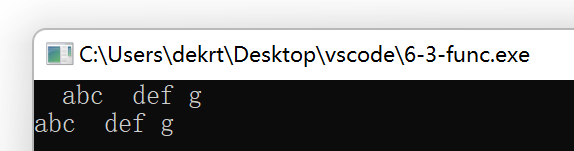


图6-11 程序设计题4的程序运行结果示意图

（5）编写8个任务函数，一个scheduler调度函数和一个execute执行函数。仅在main函数中调用scheduler函数，scheduler函数要求用最快的方式调度执行用户指定的任务函数。

①先设计task0, task1, task2, task3, task4, task5, task6, task7共8个任务函数，每个任务函数的任务就是输出该任务被调用的字符串。例如，第0个任务函数输出“task0 is called!”，第1个任务函数输出“task1 is called!”，以此类推。

②scheduler函数根据键盘输入的数字字符的先后顺序，一次调度选择对应的任务函数。例如，输入：1350并回车，则scheduler函数一次调度选择task1, task3, taks5, task0，然后以函数指针数组和任务个数为参数将调度选择结果传递给execute函数并调用execute函数。

③execute函数根据scheduler函数传递的指针数组和任务个数为参数，按照指定的先后顺序依此调用执行选定的任务函 数。

例如，当输入13607122并回车，程序运行结果如下：

task1 is called!

task3 is called!

task6 is called!

task0 is called!

task7 is called!

task1 is called!

task2 is called!

task2 is called!

1) 解题思路：

1.在main()函数中调用scheduler()函数

2.读入字符串，并将其转化为数字

3.调用对应的task函数

4.结束

2）程序清单

#include <stdio.h>

#include <string.h>

#define re register

#define maxn 1005

char s[maxn] ;

int a[maxn] ;

inline void task0()

{puts("task0 is called!") ;}

inline void task1()

{puts("task1 is called!") ;}

inline void task2()

{puts("task2 is called!") ;}

inline void task3()

{puts("task3 is called!") ;}

inline void task4()

{puts("task4 is called!") ;}

inline void task5()

{puts("task5 is called!") ;}

inline void task6()

{puts("task6 is called!") ;}

inline void task7()

{puts("task7 is called!") ;}

inline void execute ( int arr[] , int cnt )

{

for( re int i = 0 ; i < cnt ; i++ )

{

if ( a[i] == 0 ) task0() ;

if ( a[i] == 1 ) task1() ;

if ( a[i] == 2 ) task2() ;

if ( a[i] == 3 ) task3() ;

if ( a[i] == 4 ) task4() ;

if ( a[i] == 5 ) task5() ;

if ( a[i] == 6 ) task6() ;

if ( a[i] == 7 ) task7() ;

}

}

inline void scheduler()

{

scanf("%s",s) ;

int cnt = strlen(s) ;

for ( re int i = 0 ; i < cnt ; i++ )

a[i] = s[i] - '0' ;

execute ( a , cnt ) ;

}

int main()

{

scheduler() ;

return 0 ;

}

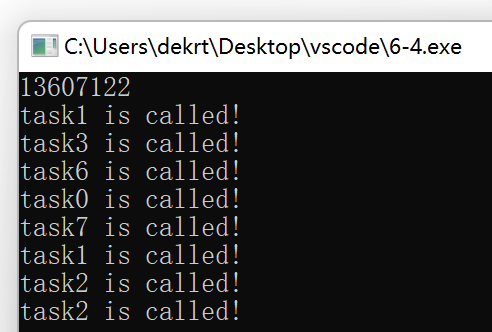


图6-12 程序设计题5的程序运行结果示意图

## 6.3 实验小结

（1）通过这次实验，我熟练掌握了指针的说明、赋值、使用。

（2）通过这次实验，我掌握了用指针引用数组的元素，熟悉指向数组的指针的使用。

（3）通过这次实验，我熟练掌握了字符数组与字符串的使用，掌握指针数组及字符指针数组的用法。

（4）通过这次实验，我掌握了指针函数与函数指针的用法。

（5）通过这次实验，我掌握了带有参数的main函数的用法。