

**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： 2019.10.19**

**网络空间安全学院**

**目 录**

**[6 表达式和标准输入输出实验 1](#_Toc404837920)**

[6.1 实验目的 1](#_Toc404837921)

[6.2 实验内容 1](#_Toc404837922)

[6.3 实验小结 1](#_Toc404837923)

# 6 表达式和标准输入与输出实验

## 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 实验内容

**6.2.1 源程序改错**

下面程序是否存在错误？如果存在，原因是什么？如果存在错误，要求在计算机上对这个例子程序进行调试修改，使之能够正确执行。

#include "stdio.h"

void main(void)

{

float \*p;

scanf("%f",p);

printf("%f\n",\*p);

}

解答：

1）该程序存在错误，在定义时，未让p指向一个有效的地址，导致其变为一个野指针。应该用malloc函数为其分配一个地址。

2）源程序清单：

#include <stdio.h>

#include <stdlib.h>

int main(void)

{

float \*p;

p=(float \*)malloc(sizeof(float));

scanf("%f",p);

printf("%f\n",\*p);

}

正确修改后的程序运行结果：

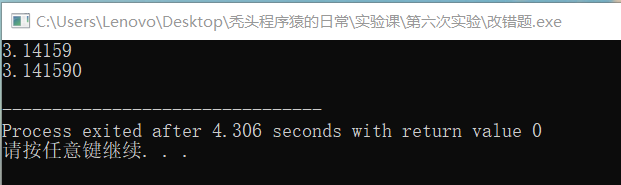


图1-1 源程序改错题的运行结果图

**1.2.1．源程序修改替换题**

（1）下面的程序通过函数指针和菜单选择来调用字符串拷贝函数或字符串连接函数，请在下划线处填写合适的表达式、语句、或代码片段来完善该程序。

#include "stdio.h"

#include "string.h"

void main(void)

{

char a[80],b[80],c[160],\*result=c;

int choice,i;

do{

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

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

printf("\t\t3 exit.\n");

printf("\t\tinput a number (1-3) please!\n");

scanf("%d",&choice);

}while(choice<1 || choice>5);

switch(choice){

case 1:

p=strcpy;

break;

case 2:

p=strcat;

break;

case 3:

goto down;

}

getchar();

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

i=0;

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

i=0;

result= (a,b);

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

down:

;

}

（2）请上机运行第（1）题程序，使之能按下面要求输出结果：（（输入）表示该数据是键盘输入数据）

1 copy string.

2 connect string.

3 exit.

input a number (1-3) please!

2 （输入）

input the first string please!

the more you learn, （输入）

input the second string please!

the more you get. （输入）

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

解答：

(1)

1. 需要加入的语句如下：

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

fgets(a,80,stdin);a[strlen(a)-1]='\0';

fgets(b,80,stdin);

result=p(a,b);

1. 程序正常运行截图如图所示：

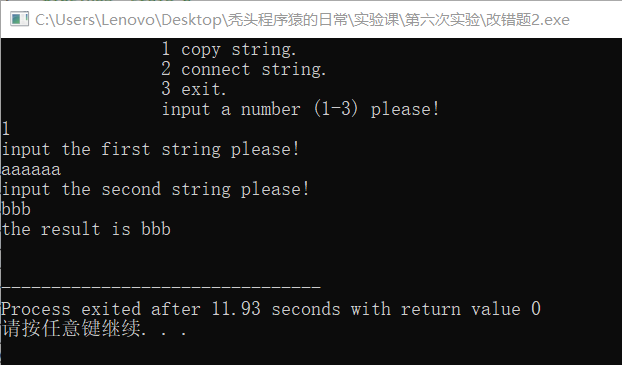


图6.2源程序完善、修改、替换题的运行结果1

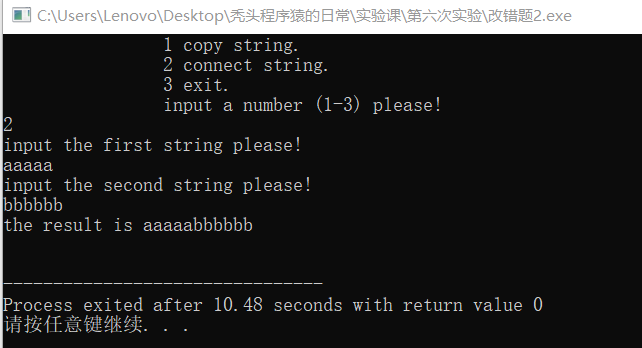
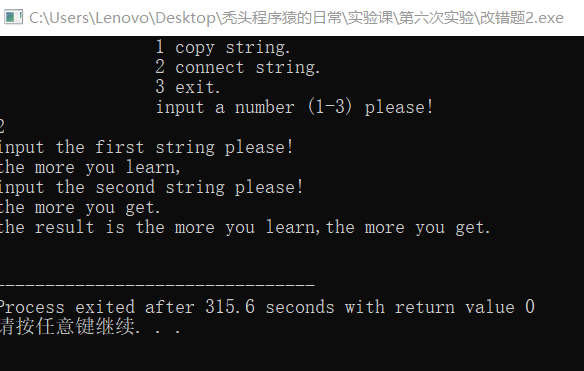


图6.3源程序完善、修改、替换题的运行结果2

(2)程序正常运行的结果如下图所示

 **图6.4源程序完善、修改、替换题的运行结果3**

6.2.3．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);

}

解答：

1. 运行结果如图所示：

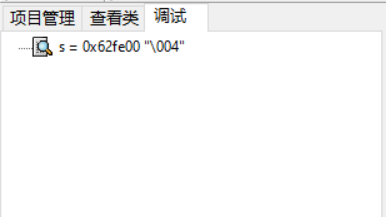


图6.5 进入watch窗口时的运行结果



图6.6 返回main时watch窗口的运行结果

1. 应增加一个指针使其指向字符串的首地址，然后返回该指针

源程序清单如下：

#include "stdio.h"

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

int 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)

{ char \*l=s;

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

;

return (l);

}

运行结果如图所示：

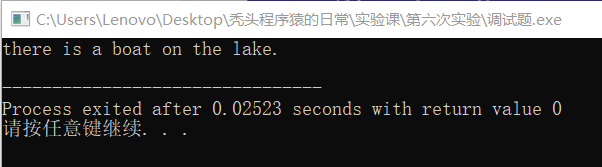


图6.7 跟踪调试题的运行结果

**1.2.4 程序设计**

（1）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。试从该长整型变量的高字节开始，依次取出每个字节的高4位和低4位并以数字字符的形式进行显示。

**解答：**

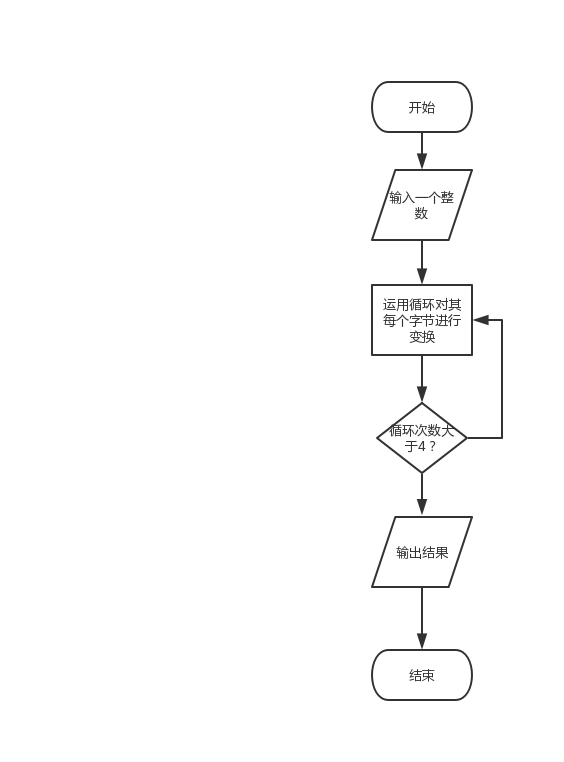
1） 算法流程如图1.1所示。

图6-8 编程题1的程序流程图

2）源程序清单

#include <stdio.h>

int main(void)

{

long t, \*p = &t;

char a,\*pp;

int n = 1, a1, a2;

scanf("%ld", &t);

pp=(char\*)p;

printf("\t\t 高四位\t\t 低四位\n");

do { a = \*pp++; a1 = a >> 4&0x0000000f;//高四位

a2 = a << 28>>28 & 0x0000000f;//低四位

printf(" 第%d 字节\t\t0x%x\t\t0x%x\n", n++, a1, a2);

}

while (n != 5);

} 3）测试

（a） 测试数据：

12345

（b） 对应测试数据的运行结果截图

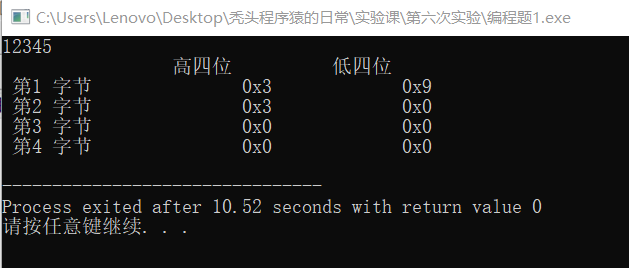


图6-9 编程题1的运行结果实例

（2）利用大小为n的指针数组指向用gets函数输入的n行，每行不超过80个字符。编写一个函数，它将每一行中连续的多个空格字符压缩为一个空格字符。在调用函数中输出压缩空格后的各行，空行不予输出。

**解答：**

1. 解题思路：

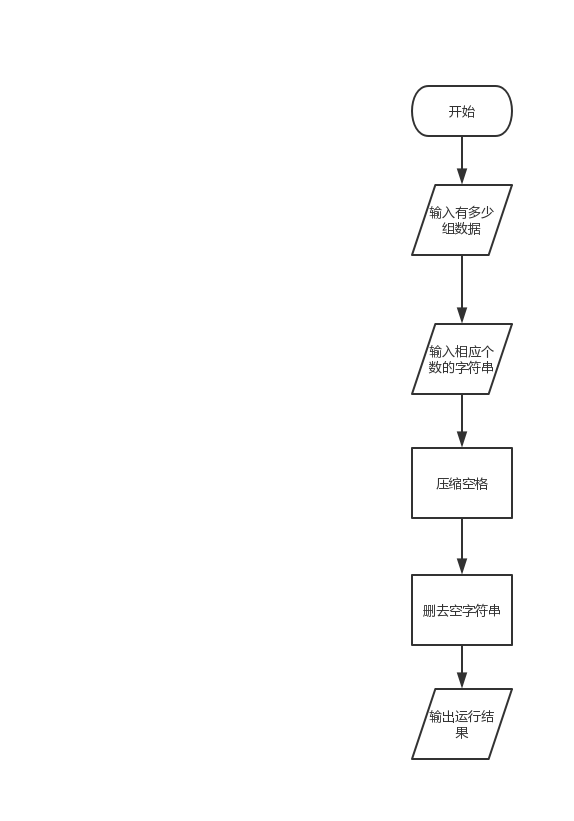


图6-10 编程题1的程序流程图

2）程序清单：

#include<stdio.h>

#include<string.h>

void zhenghe(char c[][80],int n)

{int i,x;

char \*e;

char\* d[n];

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

d[i]=&c[i][0];

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

{if(c[i][0]=='\0')

continue;

for(x=0;x<strlen(c[i]);x++)

{

inx: if ((\*(d[i]+x)==32)&&(\*(d[i]+x+1)==32))

{

x++;

goto inx;

}

printf("%c",\*(d[i]+x));

}

printf("\n");

}

e=d[i];

}

int main (void)

{int n,i;

printf("请输入要输入的行数\n");

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

char c[n][80];

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

gets(c[i]);

zhenghe(c,n);

}

3）测试

（a） 测试数据：

测试数据为3组

aaaaa bbb

ccccc ddd ee

（b）运行结果：

对应测试测试用例1的运行结果如图所示。

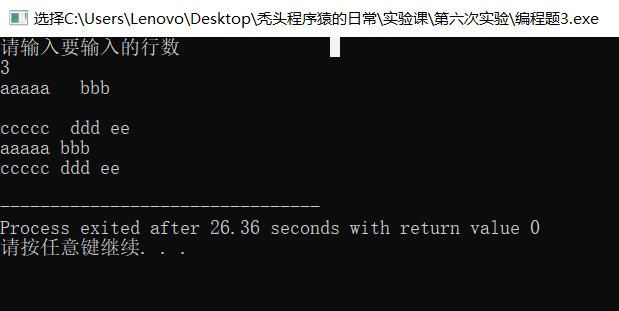


图6-12 编程题2的运行结果实例1

说明上述的运行结果与理论分析吻合，验证了程序的正确性。

**（3）**（4）设某个班有N个学生，每个学生修了M门课程（用#define定义N、M）。输入M门课程的名称，然后依次输入N个学生中每个学生所修的M门课程的成绩并且都存放到相应的数组中。编写下列函数：

a.计算每个学生各门课程平均成绩；

b.计算全班每门课程的平均成绩；

c.分别统计低于全班各门课程平均成绩的人数；

d.分别统计全班各门课程不及格的人数和90分以上（含90分）的人数。

在调用函数中输出上面各函数的计算结果。（要求都用指针操作，不得使用下标操作。）

**解答：**

1. 算法流程如图所示。

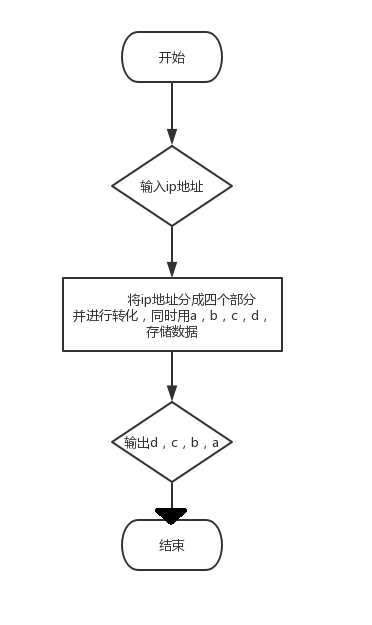


图6-13 编程题3的程序流程图

1. 源程序清单
2. #include<stdio.h>
3. #define N 5
4. #define M 3
5. int main(void)
6. {char c[M][10];
7. int a,b,i,m;
8. float e;
9. printf("请输入课程名称\n");
10. for(i=0;i<M;i++)
11. gets(c[i]);
12. int d[N][M];
13. float f[M];
14. int\* t=&d[0][0];
15. float\* x=&f[0];
16. for(a=0;a<N\*M;a++)
17. scanf("%d",t+a);
18. printf("每名学生的平均成绩是\n");
19. for(a=0;a<N;a++)
20. {e=0;
21. for(b=0;b<M;b++)
22. e+=\*(t+a\*M+b);
23. printf("学生%d的平均成绩是:%f\n",a,(e/M));
24. }
25. for(a=0;a<M;a++)
26. {e=0;
27. for(b=0;b<N;b++)
28. e+=\*(t+b\*M+a);
29. printf("科目%s的平均成绩是:%f\n",c[a],(e/N));
30. \*(x+a)=(e/N);
31. }
32. for(a=0;a<M;a++)
33. {i=0;
34. for(b=0;b<N;b++)
35. if(\*(t+b\*M+a)<\*(x+a))
36. i++;
37. printf("科目%s的低于平均人数是:%d\n",c[a],i);
38. }
39. for(a=0;a<M;a++)
40. {i=m=0;
41. for(b=0;b<N;b++)
42. {
43. if(\*(t+b\*M+a)>=90)
44. i++;
45. if(\*(t+b\*M+a)<60)
46. m++;
47. }
48. printf("科目%s的90分以上人数为:%d\n不及格人数为:%d\n",c[a],i,m);
49. }
50. }

测试

1. 测试数据：

语文 数学 嘤语

90 80 100

70 50 80

80 70 60

30 40 40

100 20 100

1. 对应测试测试用例1的运行结果如图所示。

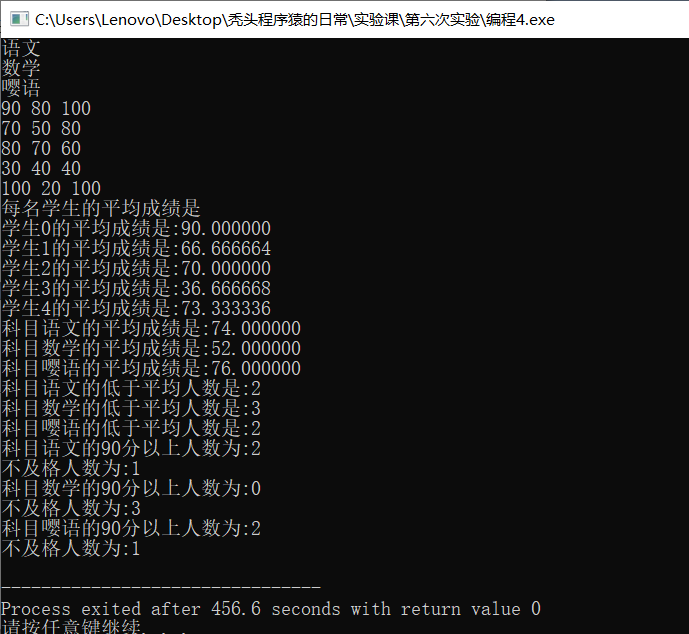


图6-14 编程题3的运行结果实例1

说明上述的运行结果与理论分析吻合，验证了程序的正确性。

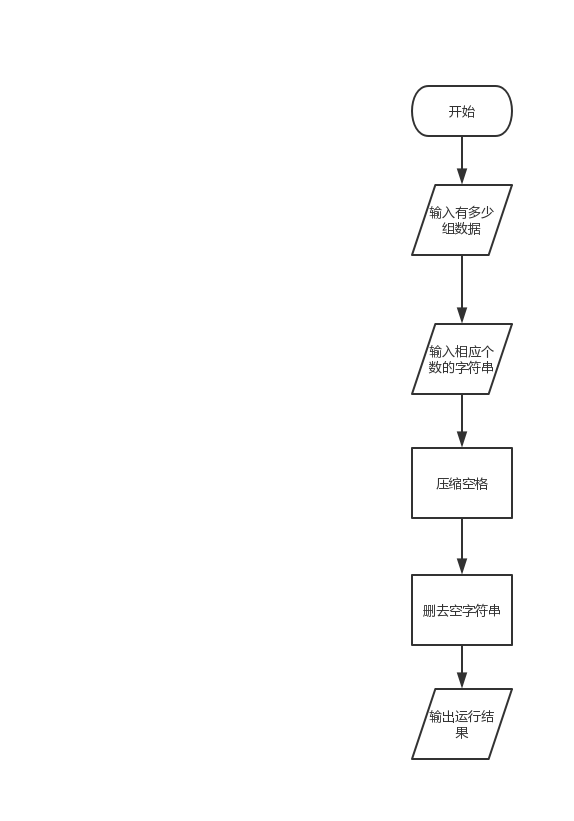
**1.2.5 选做题**

（1）设有N位整数和M位小数（N=20，M=10）的数据a,b。编程计算a+b并输出结果。

如：12345678912345678912.1234567891 + 98765432109876543210.0123456789

**解答：**

1. 解题思路：

  
图6-15 编程题3的程序流程图

2）程序清单：

#include<stdio.h>

#define N 20

#define M 10

int main(void)

{

int a[N+M+1],b[N+M+1],c[N+M+1],i,carry=0;

for(i=0;i<N+M+1;i++)

\*(c+i)=0;

i=N+M;

while(i>=0)

{

\*(a+i)=getchar();

\*(a+i)-='0';

i--;

}

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

\*(a+M+i)=\*(a+M+i+1);

getchar();

i=N+M;

while(i>=0)

{

\*(b+i)=getchar();

\*(b+i)-='0';

i--;

}

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

\*(b+M+i)=\*(b+M+i+1);

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

{

\*(c+i)=\*(a+i)+\*(b+i)+carry;

carry=(\*(c+i)-\*(c+i)%10)/10;

\*(c+i)%=10,\*(c+i)+='0';

}

\*(c+N+M)=carry+'0';

if(\*(c+N+M)!='0')

printf("%c",\*(c+N+M));

for(i=N+M-1;i>M-1;i--)

{

printf("%c",\*(c+i));

}

printf(".");

for(i=M-1;i>=0;i--)

{

printf("%c",\*(c+i));

}

} （b）运行结果：

对应测试测试用例1的运行结果如图所示。

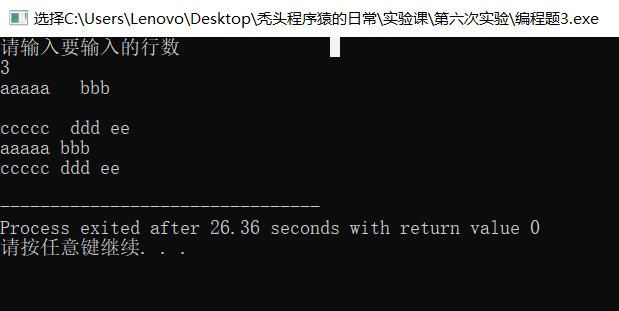


图6-16 选做题的运行结果实例1

说明上述的运行结果与理论分析吻合，验证了程序的正确性。

（2）编写使用复杂声明char \*(\*p[2])(const char \*,const char \*);的程序。

提示：p中元素可为strcmp、strstr等函数名。

**解答：**

1）程序代码：

#include<stdio.h>

#include<string.h>

int main(void)

{

char \*(\*p[2])(char \*,const char \*)={strcpy,strcat};

char a[80],b[80],c[160],\*result=c;

int choice,i;

do{

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

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

printf("\t\t3 exit.\n");

printf("\t\tinput a number (1-3) please!\n");

scanf("%d",&choice);

}while(choice<1 || choice>5);

if(choice==3)

goto down;

getchar();

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

i=0;

while((a[i]=getchar())!='\n')

i++;

a[i]='\0';

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

i=0;

while((b[i]=getchar())!='\n')

i++;

b[i]='\0';

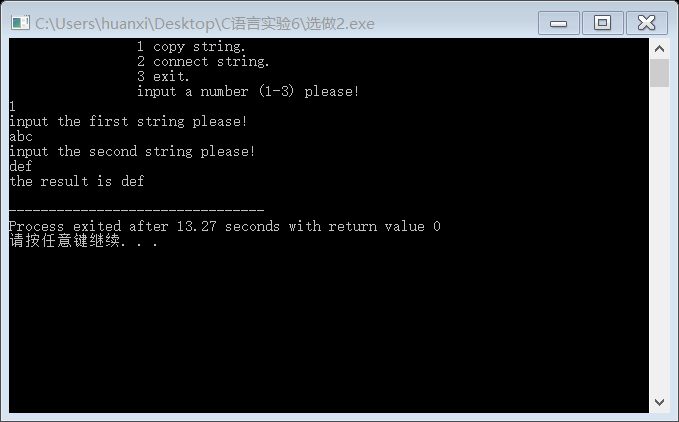
result=p[choice-1](a,b);

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

down:;

}

1. 运行结果：



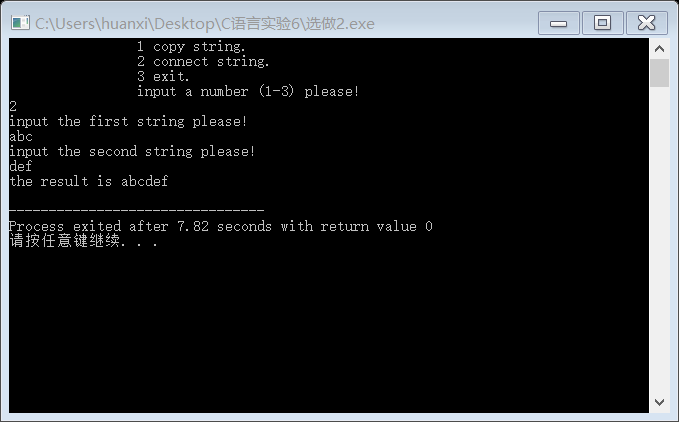


图6.17 选做题2的运行结果

## 6.3 实验小结

主要叙述实验过程中遇到的问题，如何解决的，通过分析、结果问题后的体会。  
 在这次实验的过程中，我在进行第三个编程实验时曾经遇到过不会正确调整逻辑尺对应位数的问题。最终我通过上网和查阅相关书籍解决了这一问题。

通过本次实验课，我深刻地体会到了“纸上得来终觉浅，绝知此事要躬行”这句话所蕴含的深刻道理。也许课上听完讲后我觉得自己已经掌握了相关的知识，但直到自己亲自动手编程是才发现自己的能力并没有想象中那么出色。希望我以后可以在这种活动中收获更多！

参考文献

[1] 曹计昌,卢萍,李开. C语言程序设计,北京： 科学出版社,2013

[2] 李开,卢萍,曹计昌. C语言实验与课程设计, 北京：科学出版社,2011