

函 数

【实验目的】

- (1) 掌握函数定义及调用方法,熟练掌握使用函数编写程序。
- (2) 掌握函数的嵌套调用和递归调用的方法。
- (3) 掌握函数实参和形参之间传递数据信息的方式以及返回值的概念。
- (4) 掌握单步调试进入函数和跳出函数的方法。
- (5) 掌握综合调试的方法。

【实验内容】

1. 调试样例 1

掌握单步调试进入函数和跳出函数的方法,改正下面程序中的错误。求 $1!+2!+3!+\cdots+10!=4\,037\,913.000\,000$ 。

有错误的源程序 error5_1.c:

```
#include <stdio.h>
float fac(int n);
void main()
{int i;
 double sum=0;
 for(i=1;i<=10;i++)
 sum=sum+f(i);                                /* 调试时设置断点 */
 printf("1!+2!+3!+...+10!=%f",sum);
 printf("\n");                                /* 调试时设置断点 */
}
float fac(int n)
{double result;
 if(n<0)printf("n<0,input data error!");
 else if(n==1||n==0)result=1;
 else result=n*fac(n-1);
 return n;
}
```

(1) 打开已经建立在 D 盘 C_PROGRAM 文件夹中的源程序 error5_1.c,对程序进行编译,在信息窗口中有一条警告信息(如图 1-5-1 所示):

```
'f' undefined;assuming extern returning int
```

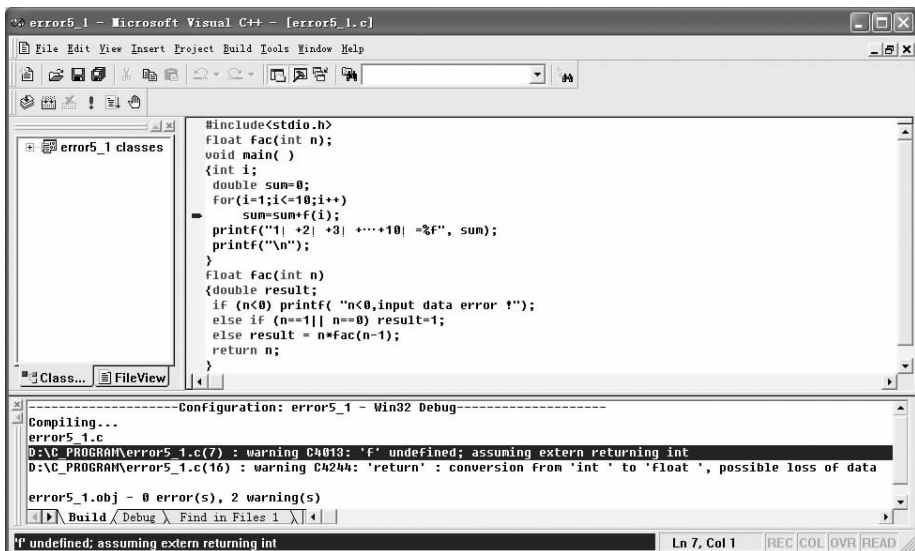


图 1-5-1 程序 error5_1.c 编译产生的信息

双击警告信息,箭头指向错误语句:“sum=sum+f(i);”。

警告提示信息指出 f 函数没有定义,根据程序分析,应将程序改正为:

```
sum=sum+ fac(i);
```

重新对程序进行编译和连接,没有错误及警告。运行程序,发现运行结果与题目不符(如图 1-5-2 所示)。

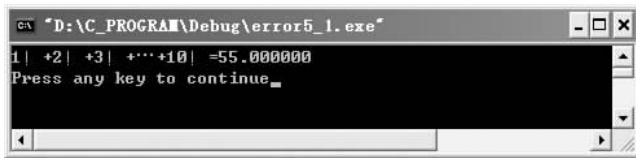


图 1-5-2 程序 error5_1.c 错误运行结果

(2) 断点设置。按照程序的注释的位置,单击 Build MiniBar/“编译微型条”工具栏中的  (Insert/Remove Breakpoint)按钮,设置断点。

(3) 调试。按照程序的注释设置断点,单击 Build MiniBar/“编译微型条”菜单中的  (Go)按钮,程序运行到第一个断点的位置(如图 1-5-3 所示)。

连续单击  (Go)按钮,在该断点位置测试,发现当 i 为 1 和 2 时,sum 计算结果正确。当 i=3 时,1!+2!+3!应为 9,而实际显示 sum=6,说明出现错误(如图 1-5-4 所示),由于计算是在函数中,说明错误出现在函数中,需要进入函数进行调试。

断点位置

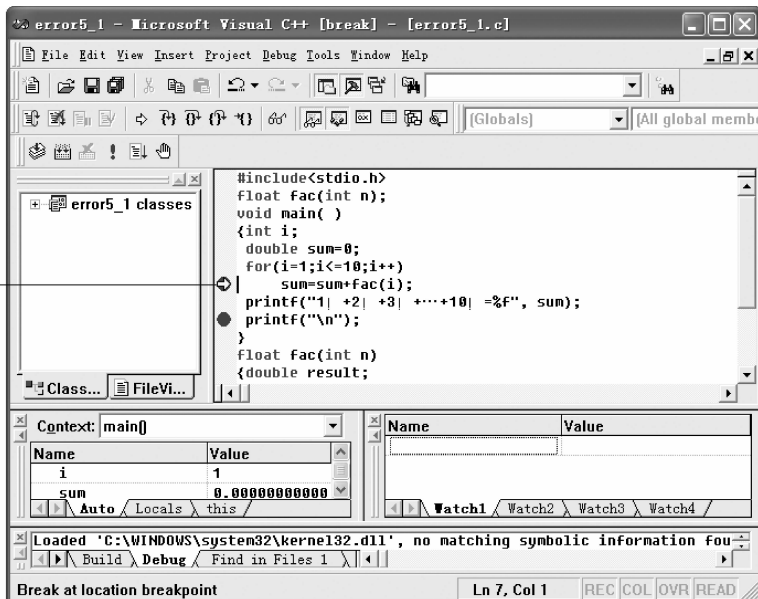


图 1-5-3 程序 error5_1.c 断点调试

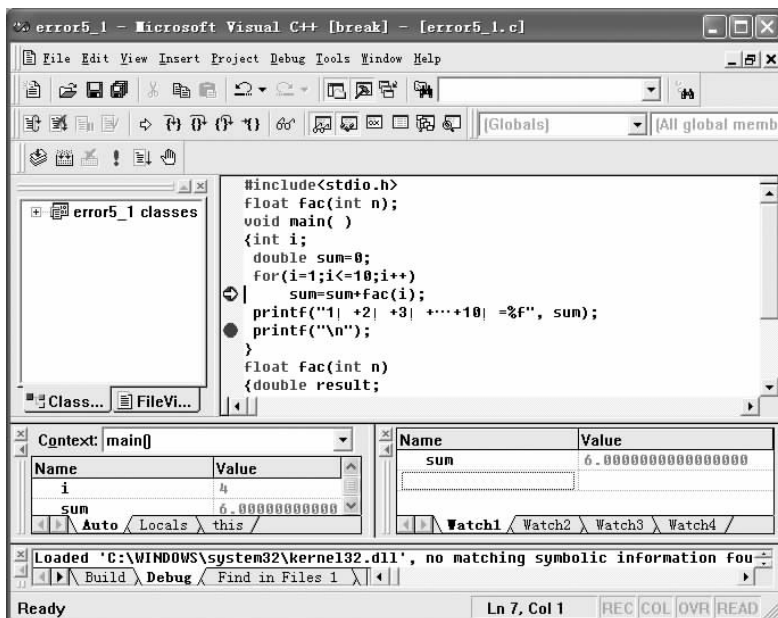


图 1-5-4 断点调试

(4) 单击 Debug/“调试”工具栏中的  (Step Into) 按钮, 程序运行进入到函数 fac 中进行调试(如图 1-5-5 所示), 窗口中的箭头表示执行到当前行。

(5) 单击  按钮 4 次, 对程序进行单步调试, 直到程序运行到函数 fac 中的光标处(如图 1-5-6 所示), 在观察窗口中观察变量 result、n 的值都正确。

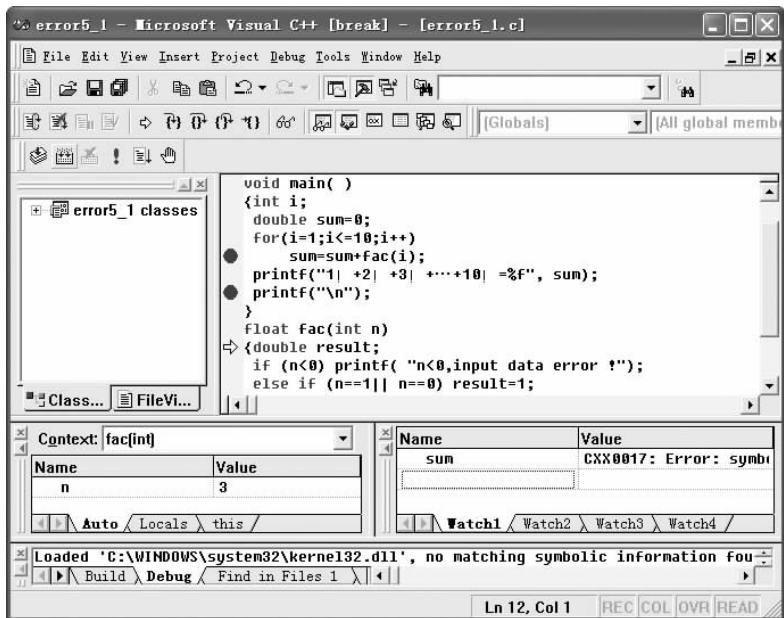


图 1-5-5 程序 error5_1.c 单步调试进入函数

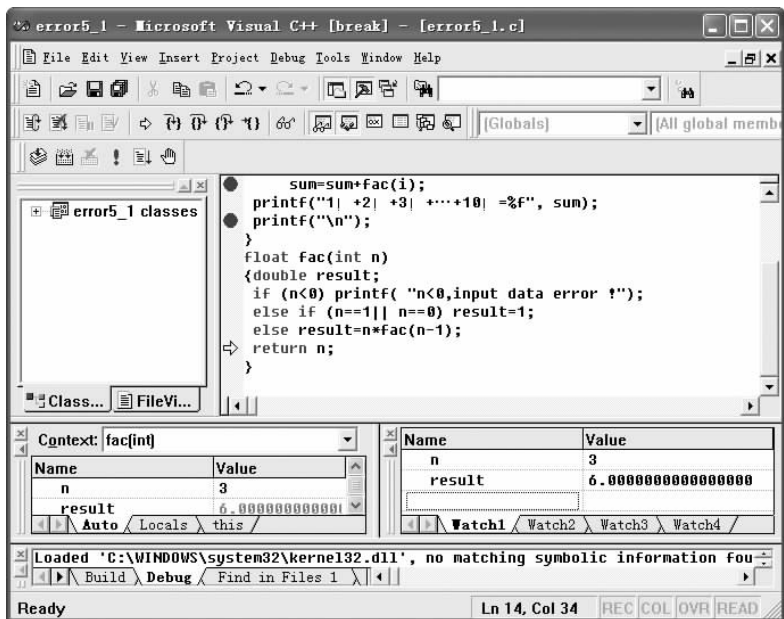


图 1-5-6 程序 error5_1.c 调试函数

分析程序,能看出函数中计算阶乘的是变量 `result`,因此,错误:“`return n;`”。

改正:“`return result;`”。

(6) 单击  (Step Out)按钮,程序返回到主程序中下面语句行:

```
sum=sum+ fac(i);
```

停止单步调试。单击 Debug/“调试”工具栏中  (Stop Debugging/“停止调试”)按钮,停止程序调试。

重新对程序进行编译,结果正确。重新断点调试,显示当 $i=3$ 时, $1!+2!+3!$ 应为 9,结果正确(如图 1-5-7 所示)。

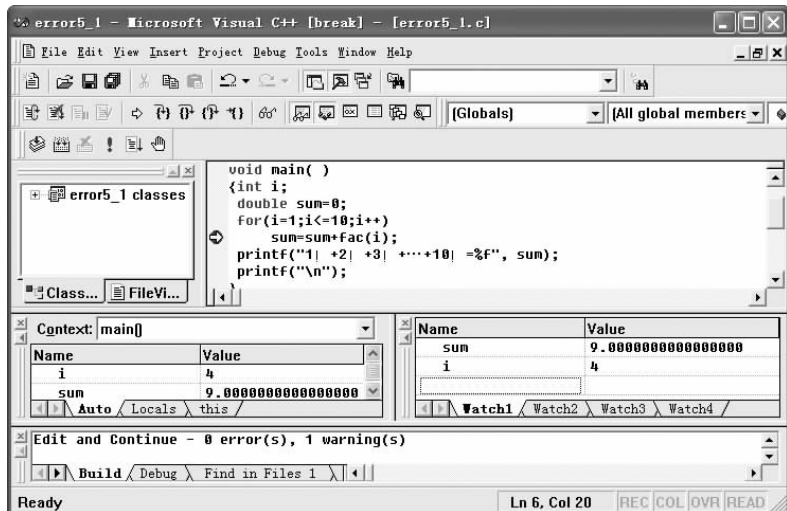


图 1-5-7 显示当 $i=3$ 时 sum 结果

(7) 继续单击  (Go)按钮,程序运行到主函数中第二个断点的位置,(如图 1-5-8 所示)。当箭头指向语句“printf("\n");”时运行窗口中显示运行结果(如图 1-5-9 所示),符合题目要求,结果正确。

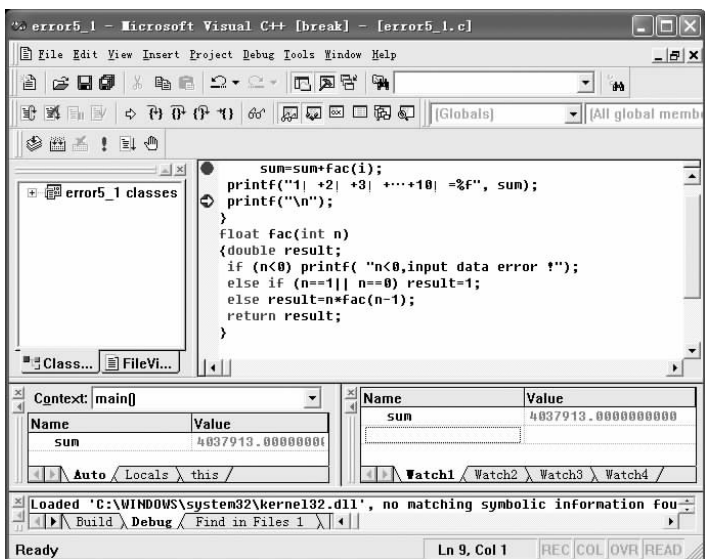


图 1-5-8 程序 error5_1.c 正确运行结果

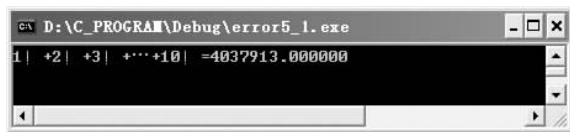


图 1-5-9 程序 error5_1.c 正确运行结果

2. 调试样例 2

掌握综合调试的方法,改正下面程序中的错误。使用选择法对 $n(n < 40)$ 个整数由小到大排序。

有错误的源程序 error5_2.c:

```
#include "stdio.h"
void sort(int array[],int n);
{int i,j,index,temp;
  for(i=0;i<n-1;i++)
    {index=i;
     for(j=i+1;j<n;j++)
       if(array[j]<array[index])index=j;
     temp=array[index];
     array[index]=array[i];
     array[i]=temp; }
}
void main()
{int i,n;
 int a[40];
 printf("enter n:");
 scanf("%d",&n);
 printf("enter the original array(split by space):\n "); /* 调试时设置断点 */
 for(i=0;i<n;i++)
   scanf("%d",&a[i]);
 sort(a,n); /* 调试时设置断点 */
 printf("the sorted array:");
 for(i=0;i<n;i++)
   printf("%3d",a[i]);
 printf("\n"); /* 调试时设置断点 */
}
```

(1) 打开已经建立在 D 盘 C_PROGRAM 文件夹中的源程序 error5_2.c,对程序进行编译,在信息窗口中有错误信息(如图 1-5-10 所示)。

从信息窗口中可以看出错误原因是缺少函数头部的函数定义,通过分析程序可知:

错误:“void sort(int array[],int n);”。

改正:“void sort(int array[],int n)”。

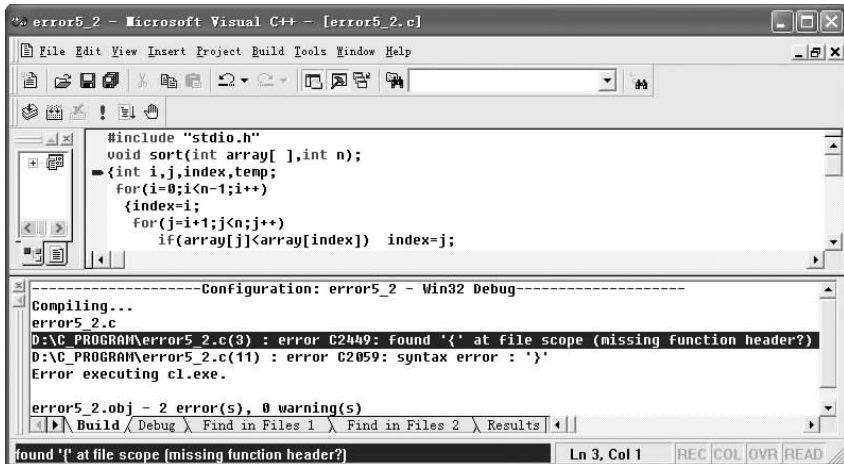


图 1-5-10 程序 error5_2.c 错误信息

重新对程序进行编译、连接,没有错误。

(2) 断点设置。按照程序的注释的位置,单击 Build MiniBar/“编译微型条”工具栏中的  (Insert/Remove Breakpoint) 按钮,设置断点。

(3) 断点调试。单击  (Go) 按钮,在运行窗口(如图 1-5-11 所示)中输入 6。程序运行到第一个断点位置。



图 1-5-11 程序 error5_2.c 调试时,输入变量 n 的值

(4) 单击  (Go) 按钮,在运行窗口中输入“4 1 5 2 6 3”。输入后程序运行到第二个断点位置,从观察窗口(如图 1-5-12 所示)中可以观察变量的值。

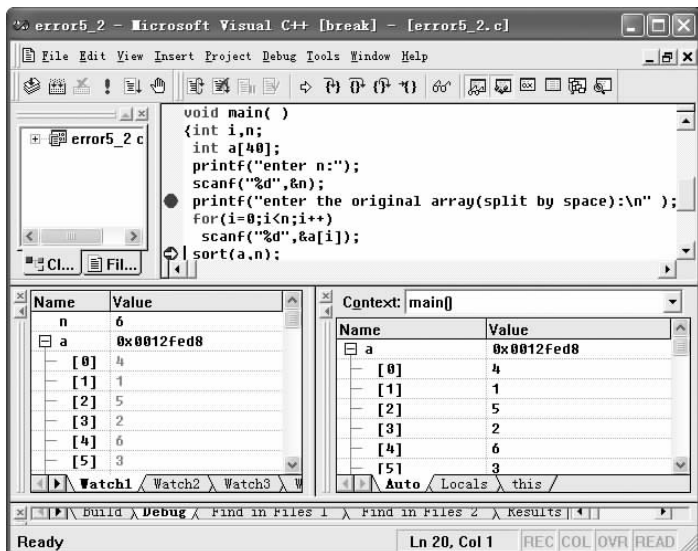


图 1-5-12 观察窗口

(5) 单步调试进入函数。单击 Debug/“调试”工具栏中的  (Step Into) 按钮,此时程序运行进入到函数 sort() 中的第一行语句。

注意: 调试到第二个断点后,若单击  (Step Over) 按钮,箭头将指向后面的“printf (“the sorted array:”);”而不进入函数。

(6) 单击  (Step Over) 按钮,在函数中进行单步调试,在变量窗口中观察变量值的变化,符合题目的要求,正确。

(7) 单步调试跳出函数。单击  (Step Out) 按钮,程序返回到主函数中,从观察窗口中可以看出数组中的元素变化,符合题意,正确。

(8) 断点调试。再次单击  (Go) 按钮,程序运行到最后一个断点,运行窗口中显示运行结果(如图 1-5-13 所示),结果正确。

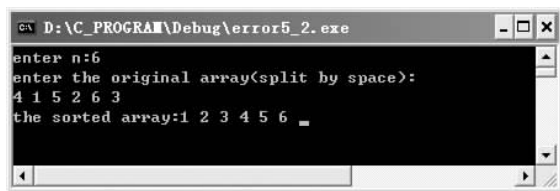


图 1-5-13 程序 error5_2.c 正确运行结果

(9) 停止调试。单击 Debug/“调试”工具栏中的  (Stop Debugging/“停止调试”)按钮,停止程序调试。

3. 程序填空题

对被调函数的声明。

```
#include "stdio.h"

void main()
{
    _____; /* 对被调函数的声明 */
    float a,b,c;
    scanf("%f,%f",&a,&b);
    c=sub(a,b);
    printf("sub is %f\n",c);
}

float sub(float x,float y)
{
    float z;
    z=x-y;
    return (z);
}
```

4. 程序修改题

模仿调试样例 1,使用单步调试进入函数和跳出函数的方法改正下面程序中的错误。

程序的功能是将 $n \times n$ 矩阵转置。

有错误的源程序 error5_3.c:

```
#include "stdio.h"
#define N 2
void convert(int arr[N][N])
{   int i,j,temp;
    for(i=0;i<N;i++)
        for(j=0;j<i;j++)
            {temp=arr[i][j]; arr[i][j]=arr[j][i]; arr[j][i]=temp;}
}
void main()
{   int i,j,array[N][N];
    printf("enter the original array(split by space):\n" );
    for(i=0;i<N;i++)
    {   for(j=0;j<N;j++)
        scanf("%d",&array[i][j]);
        printf("\n");
    }
    convert(arr);
    printf("the converted array:\n");
    for(i=0;i<N;i++)
    {   for(j=0;j<N;j++)
        printf("%d",array[i][j]);
        printf("\n");
    }
}
```

5. 程序设计题

- (1) 已知某个学生 5 门课程的成绩,求平均成绩。
- (2) 使用递归法求 $n!$ 。
- (3) 使用递归法编程。编号依次从 1 到 n 的 n 个人按照从小到大报数,数都相差 3,并且第一个人的报数是 7,问第 n 个人的报数是多少?

【实验结果和分析】

- (1) 将 C 语言源程序、运行结果写在实验报告中。
- (2) 分析源程序和运行结果,并将遇到的问题和解决问题的方法写在实验报告中。