

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级： 网安2003班**

**学 号： U202012068**

**姓 名： 叶礼亮**

**指导教师： 余林琛**

**报告日期： 2020/10/17**

**网络空间安全学院**

**目 录**

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

[1.1 实验目的 1](#_Toc404837921)

[1.2 实验内容 1](#_Toc404837922)

[1.3 实验小结 1](#_Toc404837923)

[**2 流程控制实验 2**](#_Toc404837924)

[2.1 实验目的 1](#_Toc404837921)

[2.2 实验内容 1](#_Toc404837922)

[2.3 实验小结 1](#_Toc404837923)

[**3 函数与程序结构实验 3**](#_Toc404837929)

[3.1 实验目的 1](#_Toc404837921)

[3.2 实验内容 1](#_Toc404837922)

[3.3 实验小结 1](#_Toc404837923)

[**4 编译预处理实验 4**](#_Toc404837934)

[4.1 实验目的 1](#_Toc404837921)

[4.2 实验内容 1](#_Toc404837922)

[4.3 实验小结 1](#_Toc404837923)

[**5 数组实验 5**](#_Toc404837938)

[5.1 实验目的 1](#_Toc404837921)

[5.2 实验内容 1](#_Toc404837922)

[5.3 实验小结 1](#_Toc404837923)

[**6 指针实验 6**](#_Toc404837943)

[6.1 实验目的 1](#_Toc404837921)

[6.2 实验内容 1](#_Toc404837922)

[6.3 实验小结 1](#_Toc404837923)

[**7 结构与联合实验 7**](#_Toc404837948)

[7.1 实验目的 1](#_Toc404837921)

[7.2 实验内容 1](#_Toc404837922)

[7.3 实验小结 1](#_Toc404837923)

[**8 文件实验 8**](#_Toc404837953)

[8.1 实验目的 1](#_Toc404837921)

[8.2 实验内容 1](#_Toc404837922)

[8.3 实验小结 1](#_Toc404837923)

[**参考文献 9**](#_Toc404837957)

# 4 编译预处理实验

## 4.1 实验目的

（1）掌握文件包含、宏定义、条件编译和assert宏的使用；

（2）练习使用集成开发环境中的调试功能：单步执行、设置断点、观察变量值。

（3）熟悉多文件编译技术

## 4.2 实验内容

**4.2.1 程序改错**

下面是用宏来计算平方差、交换两数的源程序.在这个源程序中存在若干错误，要求对该程序进行调试修改，使之能够正确完成指定任务。

/\*实验4-1改错与跟踪调试题程序：计算平方差、将换两数\*/

1 #include<stdio.h>

2 #define SUM a+b

3 #define DIF a-b

4 #define SWAP(a,b) a=b,b=a

5 int main()

6 {

7 int a,b;

8 printf("Input two integers a, b:");

9 scanf("%d%d", &a,&b);

10 printf("\nSUM=%d\n the difference between square of a and square of b is:%d",SUM, SUM\*DIF);

11 SWAP(a,b);

12 printf("\nNow a=%d,b=%d\n",a,b);

13 return 0;

14 }

**解答：**

（1）错误修改：

1) 第4行不能直接交换两个数，正确形式为：

#define SWAP(a,b) a=a^b,b=a^b,a=a^b

（2）错误修改后运行结果：

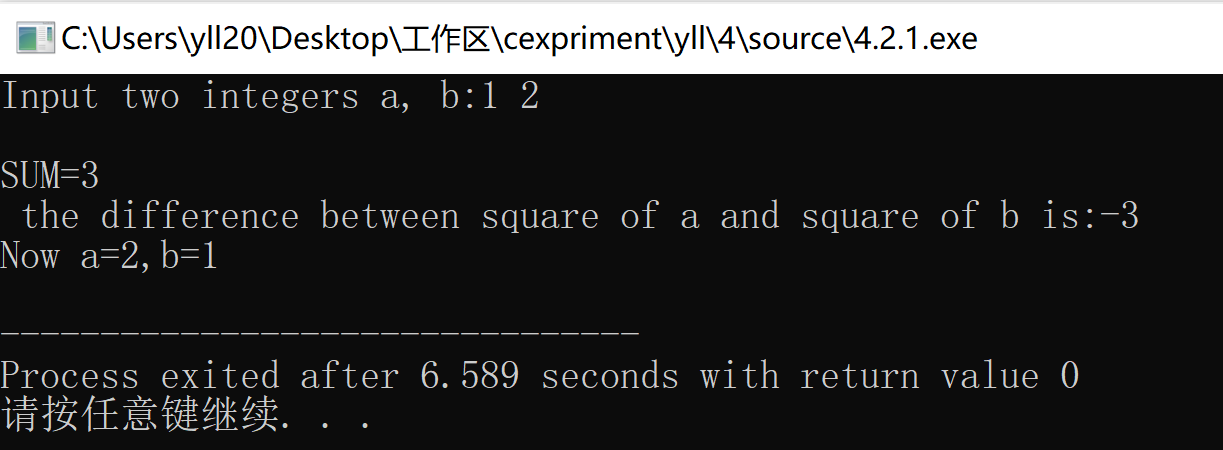
****

图4-1 改错题1的运行结果

**3.2.2 程序修改替换题**

下面是用函数实现求三个数中最大数、计算两浮点数之和的程序。在这个源程序中存在若干语法和逻辑错误。

要求：（1）对这个例子程序进行调试修改，使之能够正确完成指定任务；

（2）用带参数的宏替换函数max，来实现求最大数的功能。

/\*实验4-2程序修改替换题程序\*/

#include<stdio.h>

int main(void)

{

int a, b, c;

float d, e;

printf("Input three integers:");

scanf("%d %d %d",&a,&b,&c);

printf("\nThe maximum of them is %d\n",max(a,b,c));

printf("Input two floating point numbers:");

scanf("%f %f",&d,&e);

printf("\nThe sum of them is %f\n",sum(d,e));

return 0;

}

int max(int x, int y, int z)

{

int m=z;

if (x>y)

if(x>z) m=x;

else

if(y>z) m=y;

return m;

}

float sum(float x, float y)

{

return x+y;

}

**解答：**

替换后的程序如下所示：

/\*实验4-2程序修改替换题程序\*/

#include<stdio.h>

#define max(a,b,c) a>b?(a>c?a:c):(b>c?b:c) //求三个数的最大值

float sum(float x, float y); //函数在使用之前应该先声明

int main(void)

{

int a, b, c;

float d, e;

printf("Input three integers:");

scanf("%d %d %d",&a,&b,&c);

printf("\nThe maximum of them is %d\n",max(a,b,c));

printf("Input two floating point numbers:");

scanf("%f %f",&d,&e);

printf("\nThe sum of them is %f\n",sum(d,e));

return 0;

}

float sum(float x, float y)

{

return x+y;

}

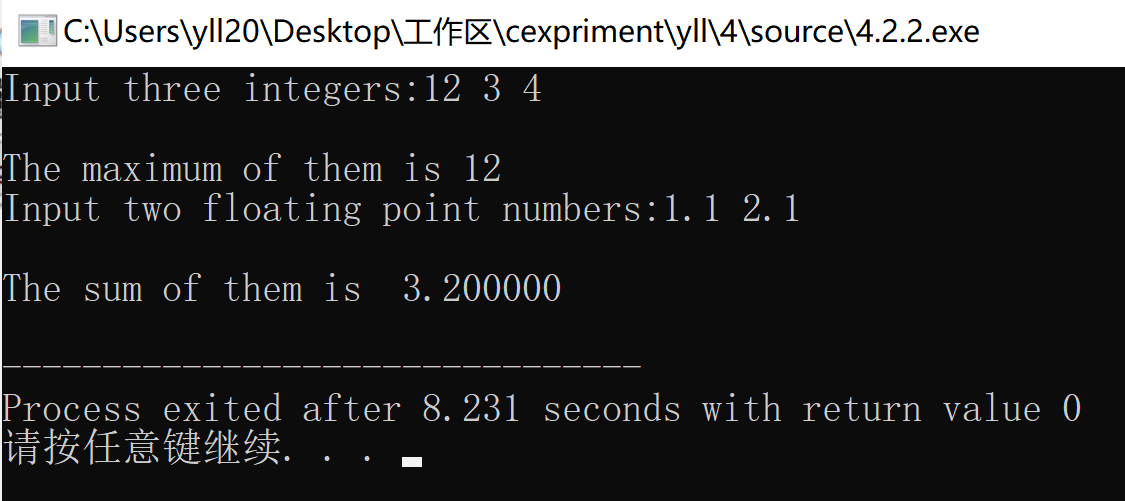


图4-2 程序替换题1的运行结果

**4.2.3 跟踪调试题**

下面程序利用R计算圆的面积s，以及面积s的整数部分。现要求：

（1）修改程序，使程序编译通过且能运行；

（2）单步执行。进入函数integerl\_fraction时，watch窗口中x为何值？在返回main时, watch窗口中i为何值？

（3）修改程序，使程序能输出面积s值的整数部分（要求四舍五入），不会输出错误信息assertion failed。

/\*实验4-3跟踪调试题程序利用R计算圆的面积s\*/

#define R

int main(void)

{

float r, s;

int s\_integer=0;

printf ("Input a number: ");

scanf("%f",&r);

#ifdef R

s=3.14159\*r\*r;

printf("Area of round is: %f\n",s);

s\_integer=integer\_fraction(s);

assert((s-s\_integer)<0.5);

printf("The integer fraction of area is %d\n", s\_integer);

#endif

return 0;

}

int integer\_fraction(float x)

{

int i=x;

return i;

}

**解答：**

1. 如图3-4所示。

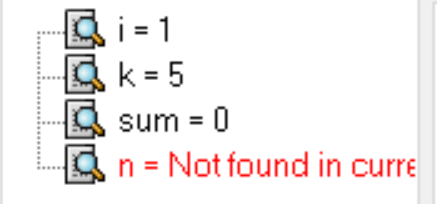


图3-4 跟踪调试题的变量查看

1. 变量值如图3-4所示。返回后停留在printf("i=%d\tthe sum is %ld\n",i,sum);如图3-5所示。

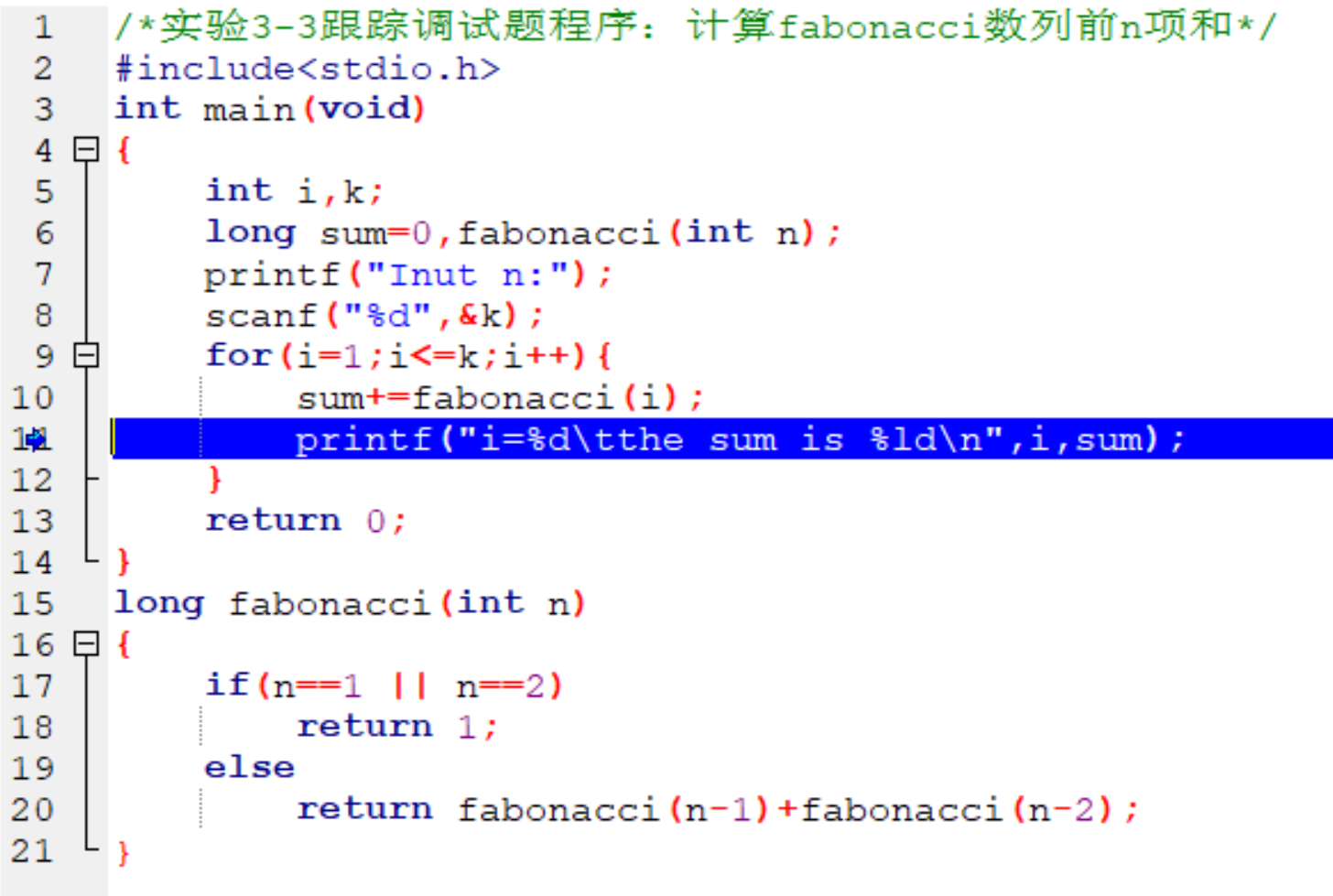


图3-5 跟踪调试题的单步调试

1. 跳转到fabonacci函数的开头。如图3-6所示。

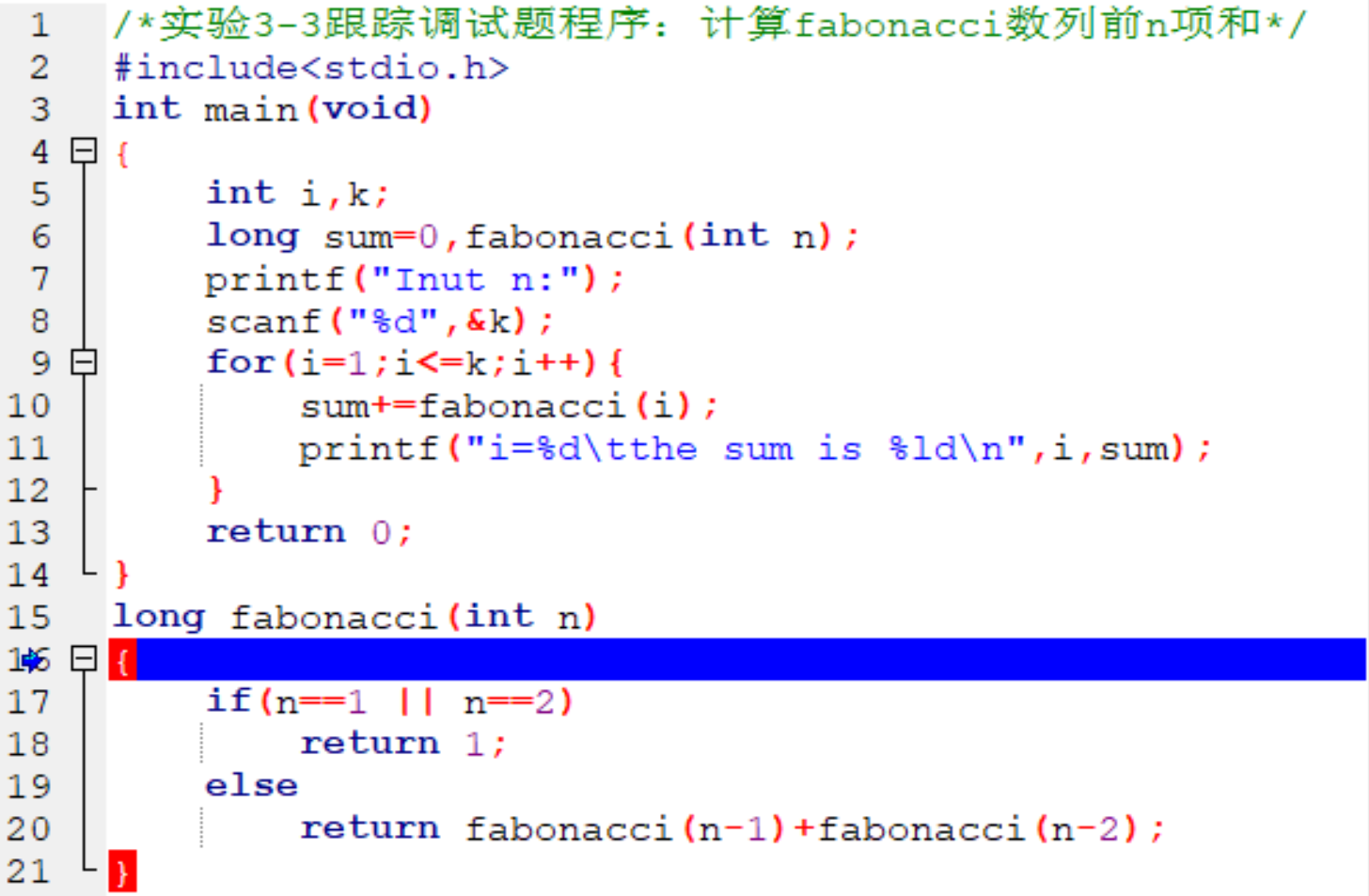


图3-6 跟踪调试题的单步调试

1. 递归函数不断调用自本身并等待最后一层调用返回结果后一层一层返回。最后直接返回到下一行printf。
2. sum和k是main函数内声明的变量,可见域只在main函数内。

**3.2.4 程序设计**

（1）编程验证歌德巴赫猜想：一个大于等于4的偶数都是两个素数之和。要求设计一个函数对其形参n验证哥德巴赫猜想，并以“n=n1+n2”的形式输出结果。例如：n=6，输出“6=3+3”。main函数循环接收从键盘输入的整数n，如果n是大于或等于4的偶数，调用上述函数进行验证。

**解答：**

1） 算法流程如图3-6所示。



图3-6 程序设计题1的程序流程图

2）源程序清单

#include <stdio.h>

#include <math.h>

int is\_veg(int n);

int main(void)

{

int num, i = 2;

scanf("%d", &num);

while(!(is\_veg(i) && is\_veg(num-i))) //两个都要是质数,否则下一个

i++;

printf("%d = %d + %d", num, i, num - i);

return 0;

}

int is\_veg(int n)

{

int i, j;

for (i = 2, j = (int)sqrt(n); i <= j; i++)

if (n % i) //不能整除就继续

continue;

else

return 0;

return 1;

}

3）测试

（a） 测试数据：

56,74

（b） 对应测试数据的运行结果截图

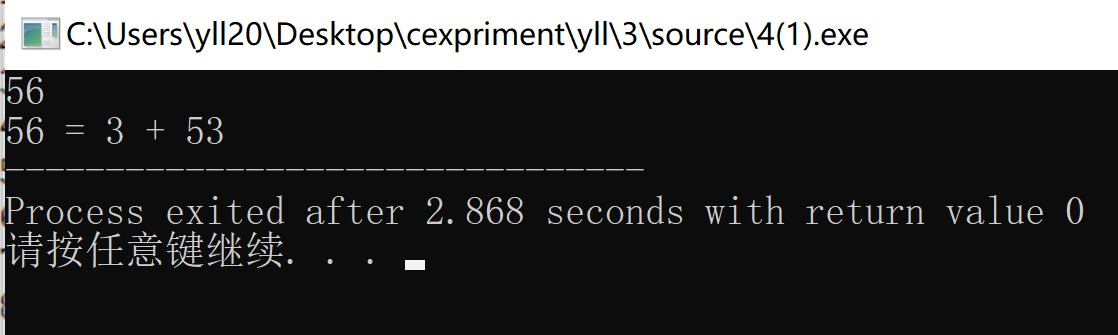


图3-7 程序设计题1的运行结果图

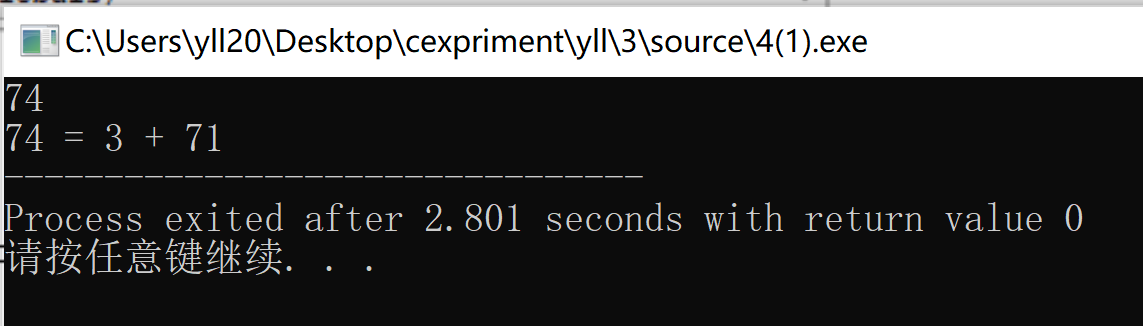


图3-8 程序设计题1的运行结果图

（2）完全数（Perfect number），又称完美数或完备数，特点是它的所有真因子（即除了自身以外的约数，包括1）之和恰好等一它本身。例如6=1+2+3，28=1+2+4+7+14等。编程寻找108以内的所有完全数。要求设计一个函数，判定形参n是否为完全数，如果是，则以n的真因子之和的形式输出结果，例如“6=1+2+3”；否则，输出“not a perfect number”，例如“5 is not a perfect number”。

在main函数中调用该函数求108以内的所有完全数。

**解答：**

1） 算法流程如图3-9所示。



图3-9 程序设计题2的程序流程图

2）源程序清单

#include <stdio.h>

#include <math.h>

int isPrime(int n);

void quickSort(int \*array, int left, int right);

int isPrime(int n)

{

if(n==2 || n==3) return 1;

if(n%6!=1 && n%6!=5) return 0;

for(int i=2;i<=sqrt(n)+1;i+=1)

{

if(n%i==0) return 0;

}

return 1;

}

void quickSort(long long \*array, int left, int right)

{

if(left < right)

{

long long pivot = array[left];

int low = left, high = right;

while(low < high)

{

while(array[high] >= pivot && low < high)

high--;

array[low] = array[high];

while(array[low] <= pivot && low < high)

low++;

array[high] = array[low];

}

array[low] = pivot;

quickSort(array, left, low - 1);

quickSort(array, low + 1, right);

}

}

int main()

{

long long mp, n = 0, i,shit;

long long b[100005];

for (i = 2; i <= 20; i++)

{

mp = pow(2,i) - 1;

if (isPrime(mp))

{

shit = (mp\*(mp+1))>>1;

if(shit<=1e8)

{

printf("%lld=",shit);

int z=0;

for(long long i=1;i\*i<=shit;i++)

{

if (shit%i==0)

{

b[z++]=i;

if(i\*i!=shit)

{

b[z++]=shit/i;

}

}

}

quickSort(b,0,z-1);

for (long i=0;i<z;i++)

{

if (i==z-1)

{

printf("\n");

break;

}

if (i==z-2)

{

printf("%lld",b[i]);

continue;

}

printf("%lld+",b[i]);

}

printf("\n");

}

}

}

return 0;

}

3）测试

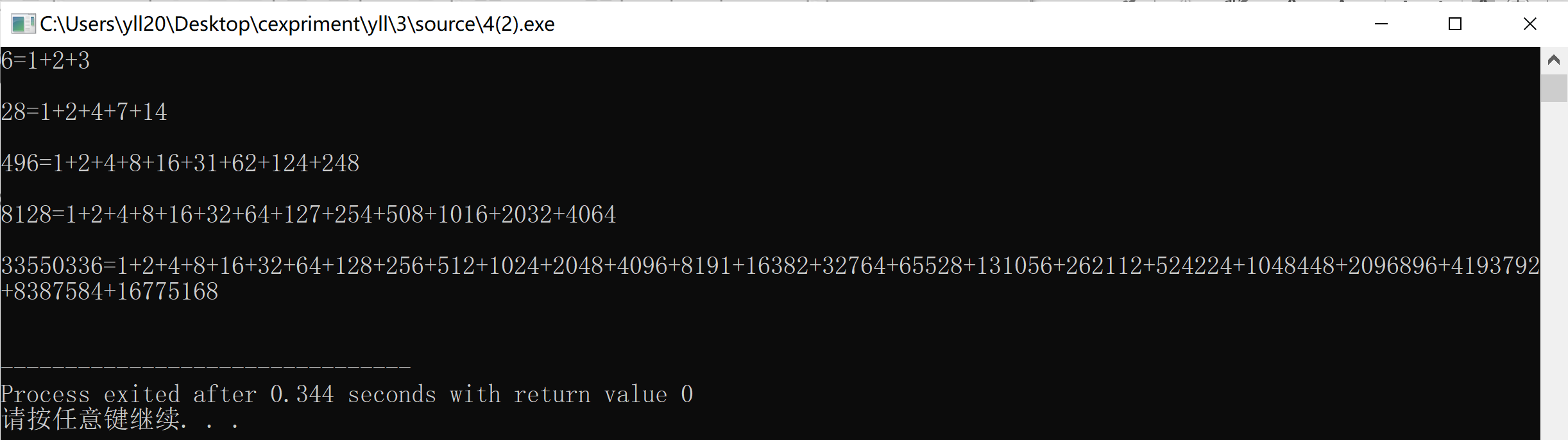


图3-10 程序设计题2的运行结果图

（3）自幂数是指一个n位数，它的每个位上的数字的n次幂之和等于它本身。水仙花数是3位的自幂数，除此之外，还有4位的四叶玫瑰数、5位的五角星数、6位的六合数、7位的北斗星数、8位的八仙数等。编写一个函数，判断其参数n是否为自幂数，如果是，则返回1；否则，返回0。main函数能反复接收从键盘输入的整数k，k代表位数，然后调用上述函数求k位的自幂数，输出所有k位自幂数，并输出相应的信息，例如“3位的水仙花数共有4个153，370，371，407”。当k=0时程序结束执行。

**解答：**

1. 算法流程如图1.1所示。



图3-11 程序设计题3的程序流程图

2）源程序清单

#include<stdio.h>

#include<math.h>

int main()

{

int cache[100];

int n;

scanf("%d",&n);

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

{

cache[i]=pow(i,n);

}

int x = 0;

for(x=pow(10,n-1);x<pow(10,n);x++)

{

int temp = x;

int sum = 0, a = 0, i;

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

{

a = temp%10; //最后一位

temp /= 10; //去掉后一位

sum += pow(a,n);

}

if(sum == x)

{

printf("%10d",x);

}

}

printf("\n");

return 0;

}

3）测试

（a） 测试数据：

3,4,5

（b） 对应测试数据的运行结果截图

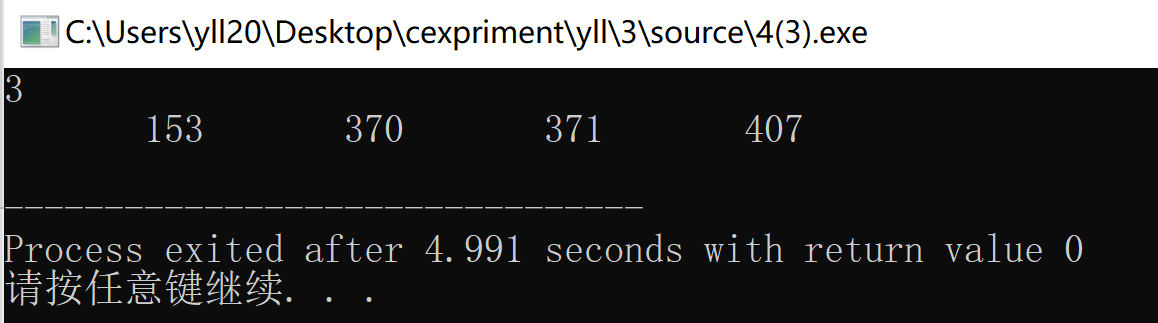


图3-12 程序设计题3的运行结果图

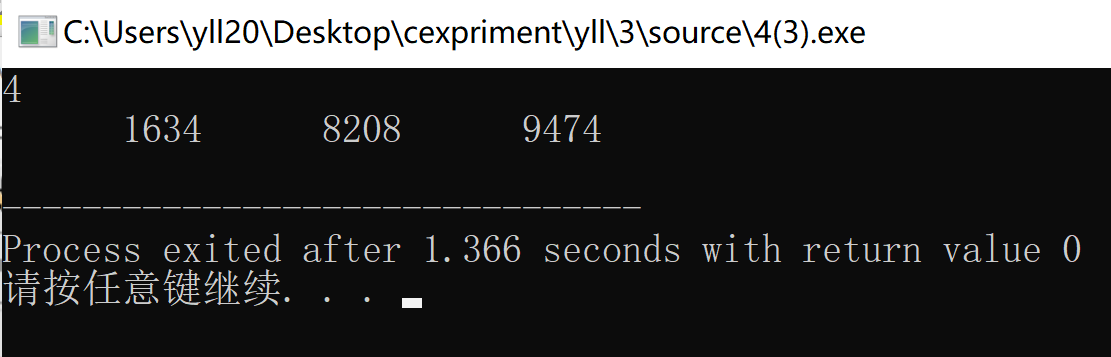


图3-13 程序设计题3的运行结果图

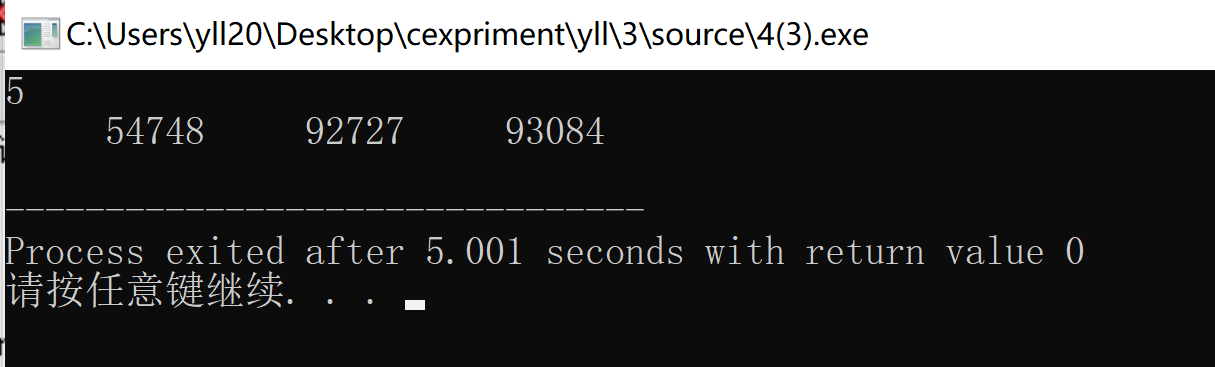


图3-14 程序设计题3的运行结果图

## 3.3 实验小结

实验中体会到了递归函数自己调用自己的思路，同时体会到了递归函数在开销上的局限性。对计算完全数的算法进行了细致的优化，提高了算法能力。

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