

**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**

[2 流程控制实验 2](#_Toc86863097)

[2.1 实验目的 2](#_Toc86863098)

[2.2 实验内容及要求 2](#_Toc86863099)

[2.2.1程序改错 2](#_Toc86863100)

[2.2.2程序修改替换 3](#_Toc86863101)

[2.2.3程序设计 6](#_Toc86863102)

[2.3 实验小结 12](#_Toc86863104)

# 2 流程控制实验

2.1 实验目的

（1）掌握复合语句、if语句、switch语句的使用，熟练掌握for、while、do-while三种基本的循环控制语句的使用，掌握重复循环技术，了解转移语句与标号语句。

（2）练习循环结构for、while、do-while语句的使用。

（3）练习转移语句和标号语句的使用。

（4）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。

## 2.2 实验内容及要求

2.2.1程序改错

下面的实验2-1程序是合数判断器（合数指自然数中除了能被1和本身整除外，还能被其它数整除的数），在该源程序中存在若干语法和逻辑错误。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2-1改错题程序：合数判断器\*/

#include <stdio.h>

int main( )

{

int i, x, k, flag = 0;

printf("本程序判断合数，请输入大于1的整数，以Ctrl+Z结束\n");

while (scanf("%d", &x) !=EOF)

{

for(i=2,k=x>>1;i<=k;i++)

if (!x%i) {

flag = 1;

break;

}

if(flag=1) printf("%d是合数", x);

else printf("%d不是合数", x);

}

return 0;

}

**解答：**

（1）错误修改：

1. 第9行的运算优先级出错，正确形式为：

if( ! (x % i ) )

1. 第13行的 if(flag=1) 语法出错，正确形式为：

if ( flag == 1 )

1. 第15行的算法出错，正确形式为：

flag = 0 ;}

(2) 错误修改后运行结果：

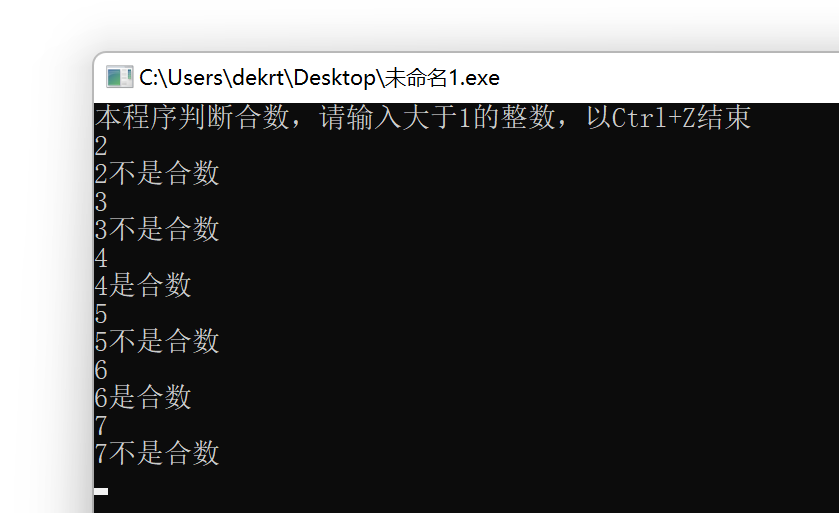


图2-1程序改错的测试用例运行结果

2.2.2程序修改替换

（1）修改实验2-1程序，将内层两出口的for循环结构改用单出口结构，即不允许使用break、goto等非结构化语句。

**解答：**

#include <stdio.h>

int main( )

{

int i, x, k, flag = 0;

printf("本程序判断合数，请输入大于1的整数，以Ctrl+Z结束\n");

while (scanf("%d", &x) !=EOF)

{

for(i=2,k=x>>1; i<=k && ( flag == 0 ) ;i++)

if (!(x%i)) flag = 1;

if(flag == 1) printf("%d是合数", x);

else printf("%d不是合数", x);

flag = 0 ;

}

return 0;

}

(2) 修改后运行结果：

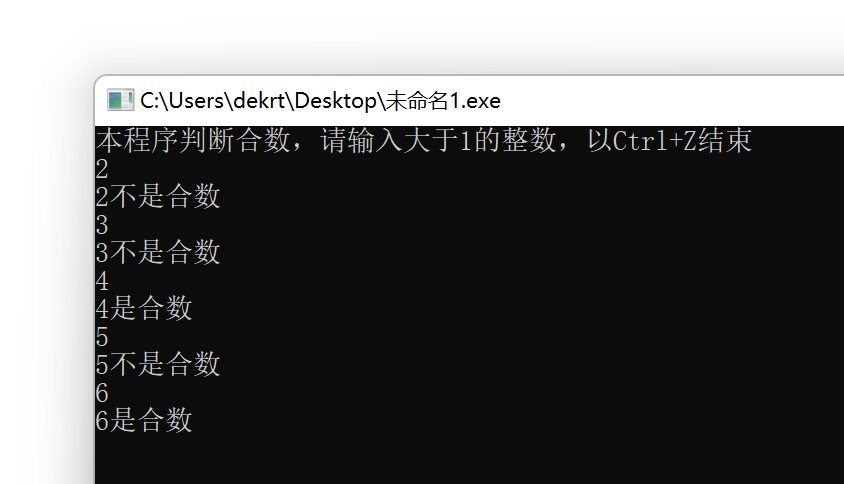


图2-2程序修改替换的测试用例运行结果

（2）修改实验2-1程序，将for循环改用do-while循环。

**解答：**

#include <stdio.h>

int main( )

{

int i, x, k, flag = 0;

printf("本程序判断合数，请输入大于1的整数，以Ctrl+Z结束\n");

do {

if(x)

{

for(i=2,k=x>>1; i<=k && ( flag == 0 ) ;i++)

if (!(x%i)) flag = 1;

if(flag == 1) printf("%d是合数\n", x);

else printf("%d不是合数\n", x);

flag = 0 ;

}

} while (scanf("%d", &x) !=EOF) ;

return 0;

}

(2) 修改后运行结果：

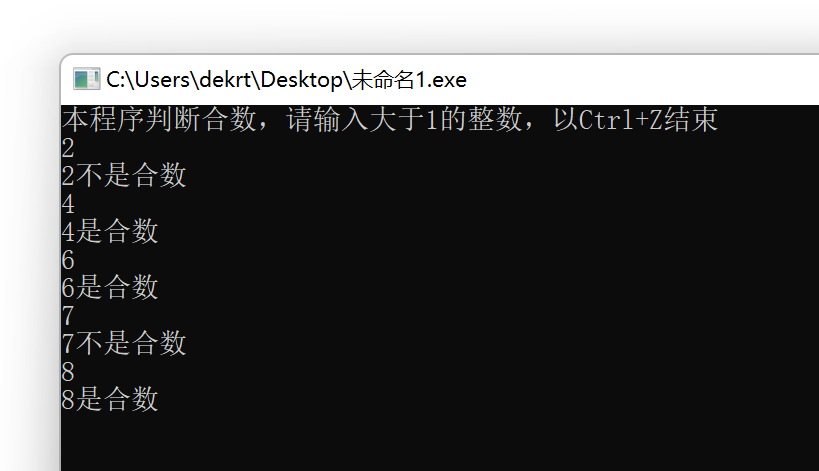
****

图2-3程序修改替换的测试用例运行结果

（3）修改实验2-1程序，将其改为纯粹合数求解器，求出所有的3位纯粹合数。一个合数去掉最低位，剩下的数仍是合数；再去掉剩下的数的最低位，余留下来的数还是合数，这样反复，一直到最后剩下一位数仍是合数，这样的数称为纯粹合数。

**解答：**

#include <stdio.h>

#include <math.h>

#define re register

int judge ( int x )

{

int tmp = sqrt(x) ;

for( re int i = 2 ; i <= tmp ; i++ )

if( x % i == 0) return 1 ;

return 0 ;

}

int main( )

{

for( re int i = 100 ; i <= 999 ; i++ )

{

if ( judge (i) && judge (i / 10) && judge (i/100) )

printf ("%d " , i ) ;

}

return 0;

}

(2) 修改后运行结果：

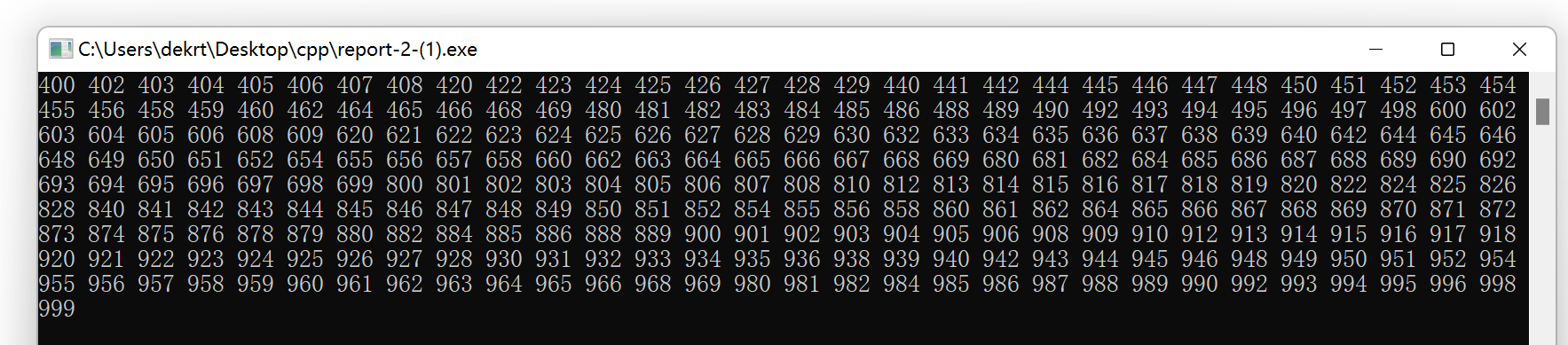


图2-4程序修改替换的测试用例运行结果

2.2.3程序设计

（1）假设工资税金按以下方法计算：x ＜ 1000元，不收取税金；1000 ≤ x ＜ 2000，收取5%的税金；2000 ≤ x ＜ 3000，收取10%的税金；3000 ≤ x ＜ 4000，收取15%的税金；4000 ≤ x ＜ 5000，收取20%的税金；5000 ≤ x，收取25%的税金。（注意税金的计算按照阶梯计税法，比如，工资为4500，那么税金=1000\*5% + 1000\*10% + 1000\*15% + 501\*20%）。编写一个程序,输入工资金额，输出应收取税金额度，要求分别用if语句和switch语句来实现。

**解答：**

/\*使用if版本\*/

#include <stdio.h>

#define re register

double x ;

int main()

{

scanf("%lf",&x) ;

int tmp = x / 1000 ;

if( tmp == 0 ) printf("收税金额为%d",0) ;

else if( tmp == 1) printf("收税金额为%.2lf", (x-1000) \* 0.05 ) ;

else if( tmp == 2) printf("收税金额为%.2lf", 50 + (x-2000) \* 0.1 ) ;

else if( tmp == 3) printf("收税金额为%.2lf", 150 + (x-3000) \* 0.15 ) ;

else if( tmp == 4) printf("收税金额为%.2lf", 300 + (x-4000) \* 0.2 ) ;

else if( tmp >= 5) printf("收税金额为%.2lf", 500 + (x-2000) \* 0.25 ) ;

return 0 ;

}

/\*使用switch版本\*/

#include <stdio.h>

#define re register

double x ;

int main()

{

scanf("%lf",&x) ;

int tmp = x / 1000 ;

switch ( tmp )

{

case 0 : printf("收税金额为%d",0) ; break ;

case 1 : printf("收税金额为%.2lf", (x-1000) \* 0.05 ) ; break ;

case 2 : printf("收税金额为%.2lf", 50 + (x-2000) \* 0.1 ) ; break ;

case 3 : printf("收税金额为%.2lf", 150 + (x-3000) \* 0.15 ) ; break ;

case 4 : printf("收税金额为%.2lf", 300 + (x-4000) \* 0.2 ) ; break ;

default : printf("收税金额为%.2lf", 500 + (x-2000) \* 0.25 ) ; break ;

}

return 0 ;

}

测试数据：

程序运行截图：



图2-5编程题1的测试用例1运行结果

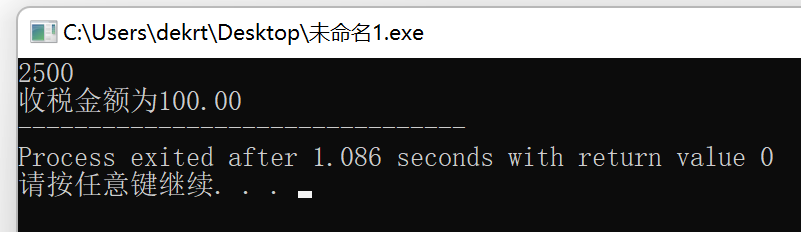


图2-6编程题1的测试用例2运行结果

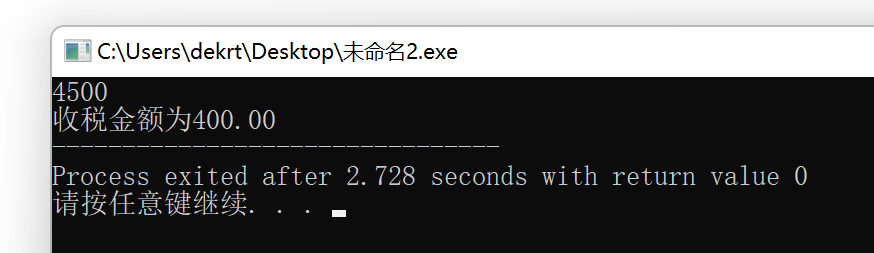


图2-7编程题1的测试用例3运行结果

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) 输入一段以!结尾的短文(最多5行,每行不超过50个字符)，要求将它复制到输出，复制过程中将每行一个以上的空格字符用一个空格代替。

1) 解题思路：

1.输入字符c，如果输入未完成则继续输入

2.如果c 为空格

2.1 如果flag为真，则继续读取字符c

2.2 如果flag为假，则flag = 1 ，并且输出字符c

3. 如果c不为空格，则flag=0,并且输出字符c

4. 结束

2）程序清单

#include <stdio.h>

char c ;

int flag = 0 ;

int main()

{

while ( scanf("%c",&c) != EOF )

{

if ( c == ' ' )

{

if ( flag ) continue ;

else

{

flag = 1 ;

putchar(c) ;

}

}

else

{

flag = 0 ;

putchar(c) ;

}

}

return 0 ;

}

3）测试

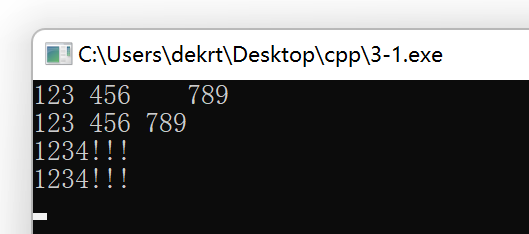


图2-8编程题2的测试用例运行结果

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）打印如下的杨辉三角形。

1 /\*第0行 \*/

1 1 /\*第1行 \*/

1 2 1 /\*第2行 \*/

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

1 6 15 20 15 6 1

1 7 21 35 35 21 7 1

1 8 28 56 70 56 28 8 1

1 9 36 84 126 126 84 36 9 1

第i行第j列位置的数据值可以由组合表示，而的计算如下：

|  |  |
| --- | --- |
|  | (i=0,1,2,…) |
|  | (j=0,1,2,3,…,i) |

根据以上公式，采用顺推法编程，输入最后一行的编号N（0<=N<=6），要求输出金字塔效果的杨辉三角形。

特别要注意空格的数目，一位数之间是3个空格，两位数之间有2个空格，3位数之间只有一个空格。第N行行首是N个空格。每行末尾是3个空格和换行。

1) 解题思路：

1.输入整数n

2.使用递推求出组合数c[i][j]

3.利用双层循环结合题意对杨辉三角进行输出

4.结束

2）程序清单

#include <stdio.h>

#define re register

#define maxn 10

int n ;

int c[maxn][maxn] ;

int main()

{

scanf("%d",&n) ;

c[0][0] = 1 ;

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

{

c[i][0] = c[i][i] = 1 ;

for( re int j = 1 ; j <= ( i >> 1 ) ; j++ )

{

c[i][j] = c[i][i-j] = c[i-1][j] + c[i-1][j-1] ;

}

}

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

{

for( re int j = 1 ; j <= n + 2\*(n-i) ; j++ )

{

printf(" ") ;

}

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

{

printf("%-4d",c[i][k]) ;

}

printf("\n") ;

}

return 0;

}

3）测试

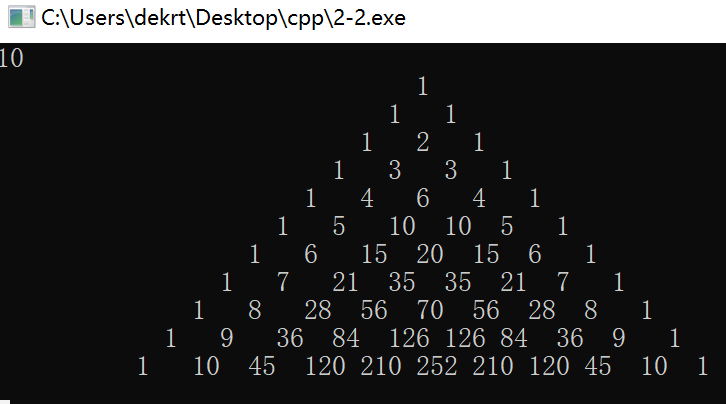


图2-9编程题3的测试用例运行结果

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。

（4）625这个数很特别，625的平方等于390625，其末3位也是625。请编程输出所有这样的3位数：它的平方的末3位是这个数本身。要求这些数字从小到大排列，每个数字单独占一行。

1) 解题思路：

1.输入字符c，如果输入未完成则继续输入

2.如果c 为空格

2.1 如果flag为真，则继续读取字符c

2.2 如果flag为假，则flag = 1 ，并且输出字符c

3. 如果c不为空格，则flag=0,并且输出字符c

4. 结束

2）程序清单

#include <stdio.h>

#define re register

int main()

{

for (re int i = 100 ; i <= 999 ; i++ )

{

if ( ( i\*i % 1000 ) == i ) printf("%d\n",i);

}

return 0;

}3）测试

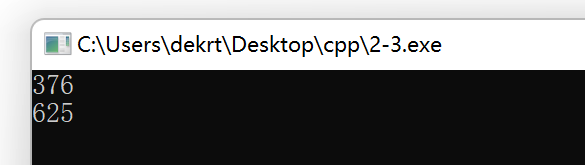


图2-10编程题4的测试用例运行结果

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。

## 2.3 实验小结

（1）通过这次试验，我掌握了复合语句、if语句、switch语句的使用，熟练掌握for、while、do-while三种基本的循环控制语句的使用，掌握重复循环技术，了解转移语句与标号语句，对它们有了更深的理解。

（2）通过这次试验，我练习了循环结构for、while、do-while语句的使用，熟练使用了他们。

（3）通过这次试验，我练习了转移语句和标号语句的使用。

（4）通过这次试验，我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，提高了debug效率。