

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级： 网络空间安全2105班**

**学 号： 网络空间安全**

**姓 名： 李翔**

**指导教师： 王同洋**

**报告日期： 2021.11.1**

**网络空间安全学院**

# 实验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. 语法错误：
2. 对于!x%i表达式应该是先取余再变为否，根据运算符优先级应该在x%i部分加上一个括号，改为!（x%i）
3. if中的flag=1，作为判断语句应该采用==，改为if（flag==1）
4. 逻辑错误：
5. 该程序一旦出现了一个合数，则flag=1，且之后并不会改变，所以应该在while循环内开头处加上

flag=0;

1. 程序源代码：

#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)

{

flag=0;

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

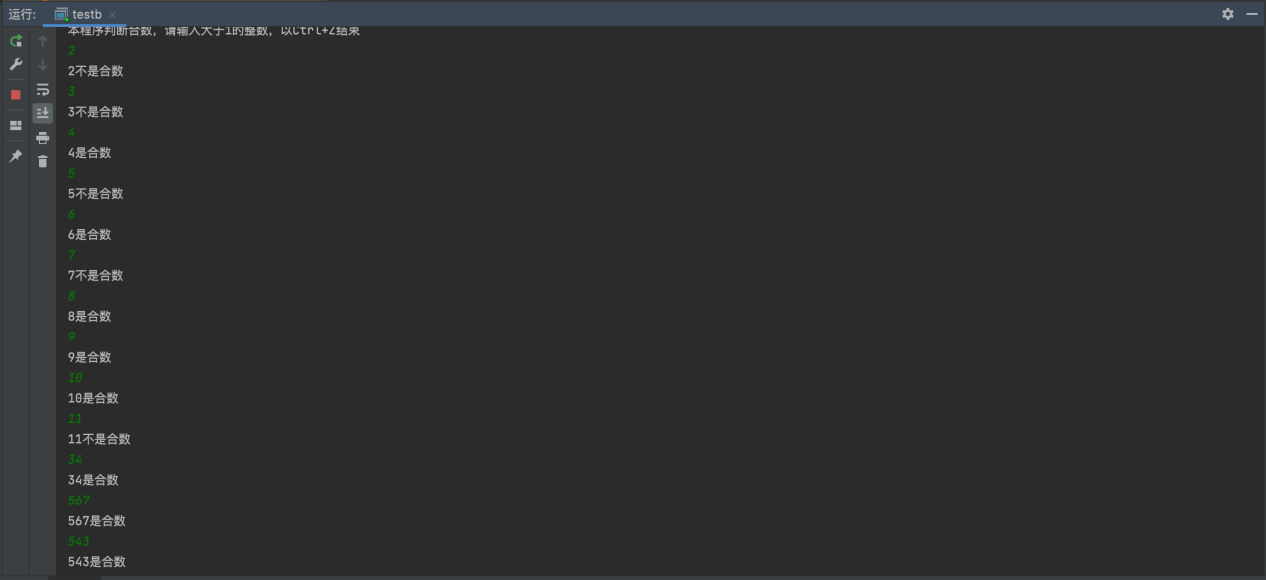
printf("%d不是合数", x);

}

return 0;

}

1. 程序运行截图



### 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。

#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)

{

flag=0;

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);

}

return 0;

}

（2）修改实验2-1程序，将for循环改用do-while循环。

#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) {

flag=0;

k=x>>1;

i=2;

do {

if(x==2)

{

break;

}

else if (!(x%i))

{

flag = 1;

break;

}

i++;

}while(i<=k);

if(flag==1)

printf("%d是合数", x);

else

printf("%d不是合数", x);

}

return 0;

}

}

1. 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。

#include <stdio.h>

int main( )

{

int i, x=100,y, k, flag;

while (x>99&&x<1000)

{

flag=0;

y=x;

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

{

if (!(x%i))

{

flag = 1;

}

}

if(flag==1)

{

y/=10;

flag=0;

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

{

if (!(y%i))

{

flag = 1;

}

}

if(flag==1)

{

y/=10;

flag=0;

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

{

if (!(y%i))

{

flag = 1;

}

}

if(flag==1)

printf("%d\n",x);

}

}

x++;

}

return 0;

}

### 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。

if语句实现

#include <stdio.h>

int main( )

{

int x;

double y;

scanf("%d",&x);

if(x<1000)

{

printf("%d",x);

}else if(x>=1000&&x<2000)

{

y=(x-1000)\*0.05;

printf("%f",y);

}else if(x>=2000&&x<3000)

{

y=1000\*0.05+(x-2000)\*0.1;

printf("%f",y);

}else if(x>=3000&&x<4000)

{

y=1000\*0.05+1000\*0.1+(x-3000)\*0.15;

printf("%f",y);

}else if(x>=4000&&x<5000)

{

y=1000\*0.05+1000\*0.1+3000\*0.15+(x-4000)\*0.2;

printf("%f",y);

}else

{

y=1000\*0.05+1000\*0.1+3000\*0.15+4000\*0.2+(x-5000)\*0.25;

printf("%f",y);

}

return 0;

}

Switch语句实现

#include <stdio.h>

int main()

{

int x;

double y;

scanf("%d",&x);

switch (x/1000)

{

case 0:

printf("%d",x);

break;

case 1:

y=(x-1000)\*0.05;

printf("%f",y);

break;

case 2:

y=1000\*0.05+(x-2000)\*0.1;

printf("%f",y);

break;

case 3:

y=1000\*0.05+1000\*0.1+(x-3000)\*0.15;

printf("%f",y);

break;

case 4:

y=1000\*0.05+1000\*0.1+3000\*0.15+(x-4000)\*0.2;

printf("%f",y);

**break;**

default:

y=1000\*0.05+1000\*0.1+3000\*0.15+4000\*0.2+(x-5000)\*0.25;

printf("%f",y);

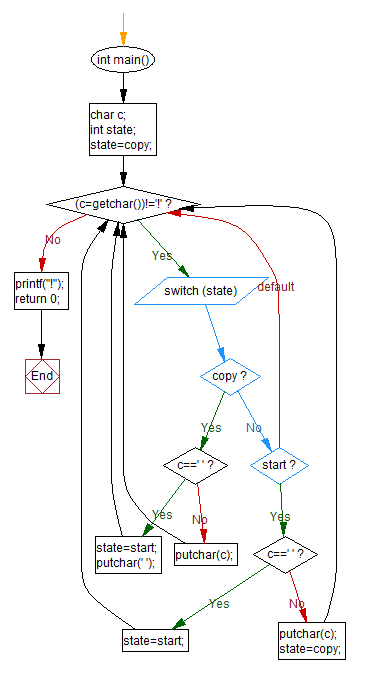
break;

}

}

### 2.2.4头歌实验

1. 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。
2. 程序流程图



1. 程序源代码

#include <stdio.h>

enum {copy,start};

int main() {

char c;

int state;

state=copy;

while((c=getchar())!='!')

{

switch (state)

{

case copy:

if(c==' ')

{

state=start;

putchar(' ');

}

else

putchar(c);

break;

case start:

if(c==' ')

state=start;

else

{

putchar(c);

state=copy;

}

break;

}

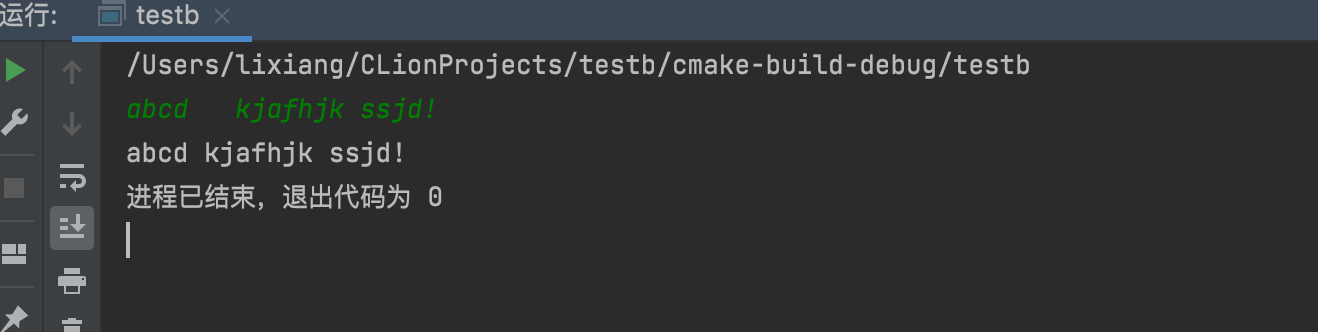
}

printf("!");

return 0;

}

1. 程序运行结果



2: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

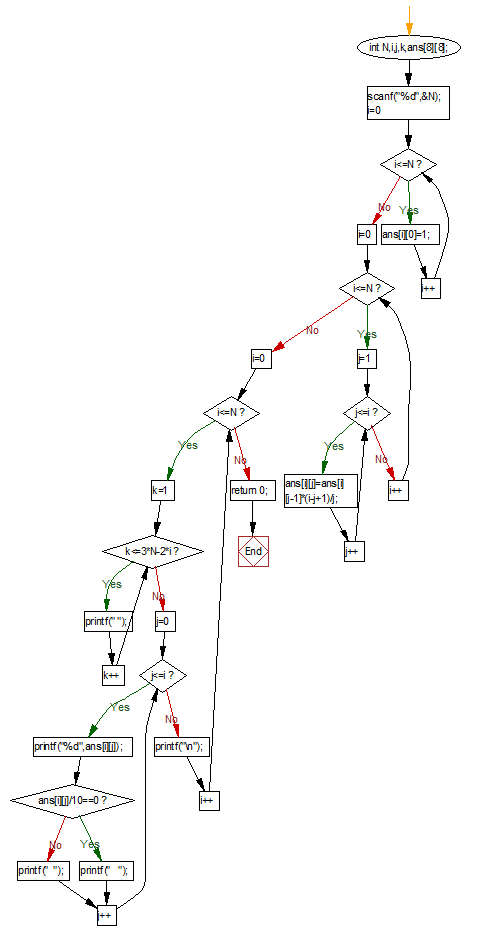
第i行第j列位置的数据值可以由组合C表示，C的计算如下：

,

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，每个数字占位4格，例如，1位数后补3个空格。第N行行首是N个空格（每向上一行，行首空格数对应增加）。每行末尾是换行符。

1. 程序流程图



1. 程序源代码

#include <stdio.h>

int main() {

int N,i,j,k,ans[N][N];

scanf("%d",&N);

for(i=0;i<=N;i++) ans[i][0]=1;

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

for(j=1;j<=i;j++) ans[i][j]=ans[i][j-1]\*(i-j+1)/j;

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

for(k=1;k<=3\*N-2\*i;k++) printf(" ");

for(j=0;j<=i;j++) {

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

if(ans[i][j]/10==0) printf(" ");

else printf(" ");

}

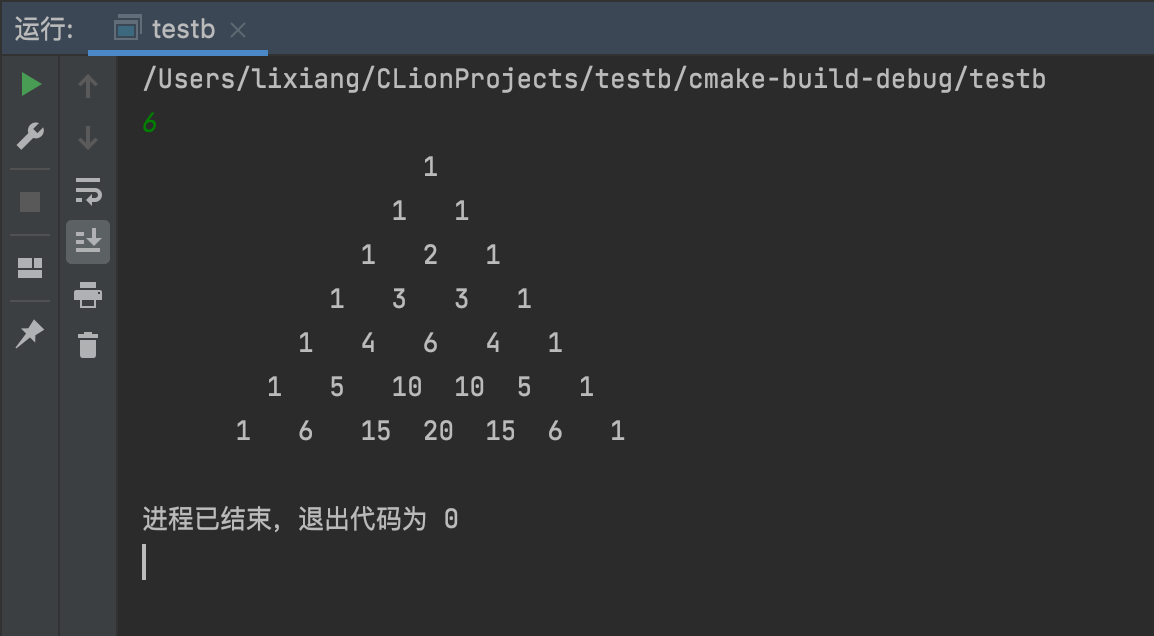
printf("\n");

}

return 0;

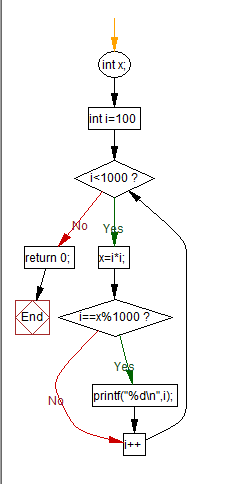
}

1. 测试



3.625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. 程序流程图



1. 程序源代码

#include <stdio.h>

int main()

{

int x;

for(int i=100;i<1000;i++)

{

x=i\*i;

if(i==x%1000)

{

printf("%d\n",i);

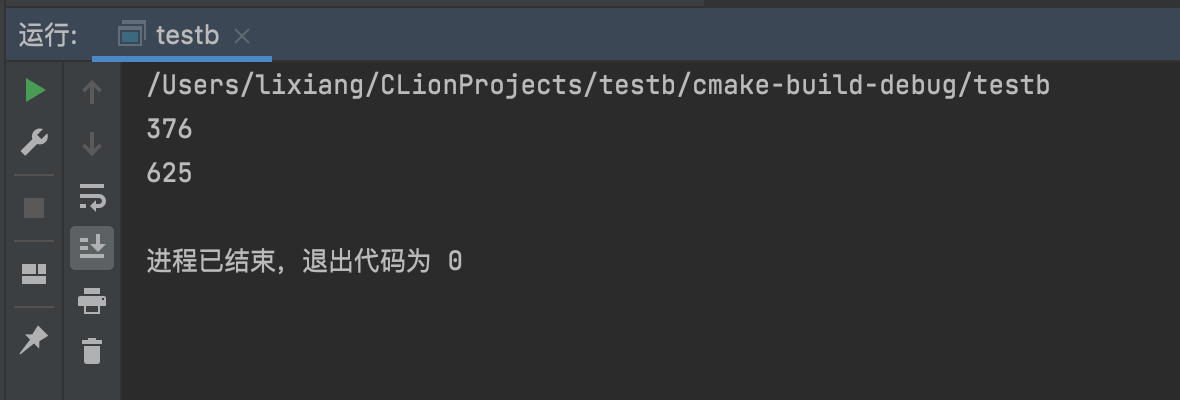
}

}

return 0;

}

1. 程序测试结果



2.3 实验小结

通过本次的实验，我主要有以下的心得体会。

使用数组存储字符串，要将四位数结果前面补0，可以用%05d的方法巧妙的解决。

心得：找出最简单的思维方式，抓大放小，明析每一个循环和每一个函数的意义，把最主要的模块化的东西先写出来，然后再去完成其中的细节。