# 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Turbo C 2.0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

## 2.2.1．源程序改错题

下面是计算s=n!的源程序，在这个源程序中存在若干语法和逻辑错误。要求在计算机上对这个例子程序进行调试修改，使之能够正确完成指定任务。例如，8！=40320。

1 #include <stdio.h>

2 void main(void)

3 {

4 int i,n,s=1;

5 printf("Please enter n:");

6 scanf("%d",n);

7 for(i=1,i<=n,i++)

8 s=s\*i;

9 printf("%d! = %d",n,s);

10 }

11 return 0;

**解答：**

（1）错误修改：

1) 第2行的void main(void)有误，正确形式为：

int main(void)

2) 第4行的int s;存在漏洞，当n比较大时，int型的位数不够，longlong型可以更好容纳，正确形式为：

int i,n; long long s=1;

3) 第6行的n前缺少取地址符，正确形式为：

scanf(“%d”,&n);

4) 第7行的标点符号错误，应把逗号改为分号，正确形式为：

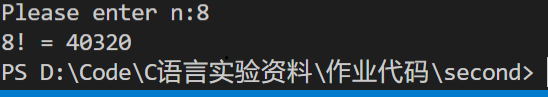
for(i=1;i<=n;i++)

5）第9行的printf应对应为：

printf("%d! = %.0lf",n,s);

6）第9，10行右花括号与return语句应交换；

（2）错误修改后运行结果：如图2-1所示



**图2-1 改错题的测试的运行结果**

2.2.2．源程序修改替换题

1. 修改第1题，分别用while和do-while语句替换for语句。

#include <stdio.h>

void main(void)

{

int i,n,s=1;

printf("Please enter n:");

scanf("%d",&n);

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

s=s\*i;

printf("%d! = %d",n,s);

return 0;

}

**解答：**

#include <stdio.h>

int main(void){

int i, n, s=1;

printf("Please enter n:");

scanf("%d", &n);

#if 0

i = 1;

while(i<=n){

s \*= i;

i++;

}

#endif

#if 1

i = 1;

do{

s \*= i;

i++;

}while(i<=n);

#endif

printf("%d! = %d", n, s);

return 0;

}

**（2）**修改第1题，输入改为“整数S”，输出改为“满足n！≥S的最小整数n”。例如输入整数40310，输出结果为n=8。

**解答：**

替换后的程序如下所示：

#include <stdio.h>

int main(void){

int i, n = 1, s=1, su = 1;

printf("Please enter s:");

scanf("%d", &s);

for(n = 1; su < s; n++){

su \*= n;

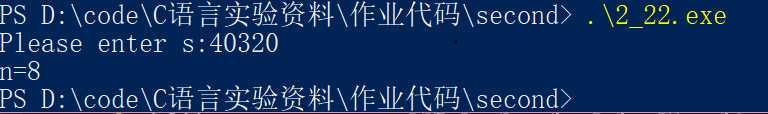
}

printf("n=%d", n-1);

return 0;

}

替换后运行结果如图2-2所示：



**图2-2 替换题的测试的运行结果**

2.2.3．编程设计题

编写并上机调试运行能实现以下功能的程序。

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；x＞5000，收取25%的税金。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：**

A）if语句

1) 解题思路：

运用if构造条件选择，x ＜ 1000，不收税金；1000 ≤ x ＜ 2000，收5%税金；2000 ≤ x ＜ 3000，收10%税金；3000 ≤ x ＜ 4000，收15%税金；4000 ≤ x ＜ 5000，收20%税金；x＞5000，收25%税金。

2) 源程序清单

#include<stdio.h>

int main(void){

double x;

int i;

scanf("%lf", &x);

#if 1

if(x < 1000)

printf("No Tax\n");

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

printf("Tax is %lf\n", 0.05\*(x-1000);

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

printf("Tax is %lf\n", 0.1\*(x-2000)+50);

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

printf("Tax is %lf\n", 0.15\*(x-3000)+150);

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

printf("Tax is %lf\n", 0.2\*(x-4000)+300);

else if(x > 5000)

printf("Tax is %lf\n", 0.25\*(x-5000)+500);

#endif

return 0;

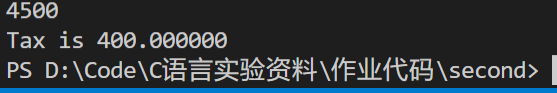
}

3) 测试

（a） 测试数据：x=4500

（b） 对应测试数据的运行结果截图

对应测试的运行结果如图2-3所示。



**图2-3 编程题1A的测试运行结果**

B）switch语句

1) 解题思路：

运用switch构造条件选择，x ＜ 1000，不收税金；1000 ≤ x ＜ 2000，收5%税金；2000 ≤ x ＜ 3000，收10%税金；3000 ≤ x ＜ 4000，收15%税金；4000 ≤ x ＜ 5000，收20%税金；x＞5000，收25%税金。

2) 源程序清单

#include<stdio.h>

int main(void){

double x;

int i;

scanf("%lf", &x);

if(x < 1000)

i = 1;

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

i = 2;

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

i = 3;

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

i = 4;

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

i = 5;

else if(x > 5000)

i = 6;

switch(i){

case 1:

printf("No Tax\n");

break;

case 2:

printf("Tax is %lf\n", 0.05\*(x-1000));

break;

case 3:

printf("Tax is %lf\n", 0.1\*(x-2000)+50);

break;

case 4:

printf("Tax is %lf\n", 0.15\*(x-3000)+150);

break;

case 5:

printf("Tax is %lf\n", 0.2\*(x-4000)+300);

break;

case 6:

printf("Tax is %lf\n", 0.25\*(x-5000)+500);

break;

}

return 0;

}

3) 测试

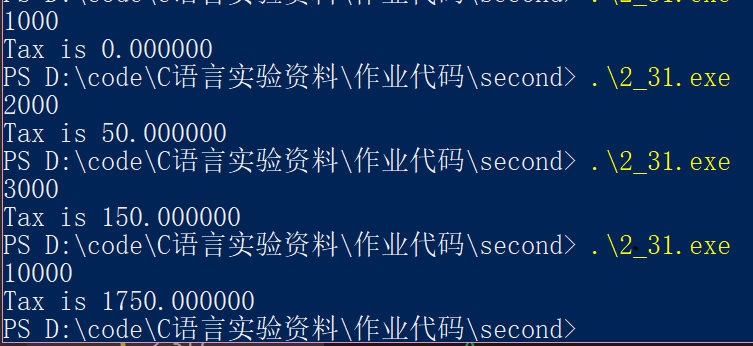
（a） 测试数据：如表2-1所示

**表2-1-2 编程题1B的测试数据**

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 程序输入 | 理论结果 | 运行结果 |
| 用例1 | 1000 | 0.000000 | 0.000000 |
| 用例2 | 2000 | 50.000000 | 50.000000 |
| 用例3 | 3000 | 150.000000 | 150.000000 |
| 用例4 | 10000 | 1750.000000 | 1750.000000 |

（b） 对应测试数据的运行结果截图

对应测试的运行结果如图2-4所示。

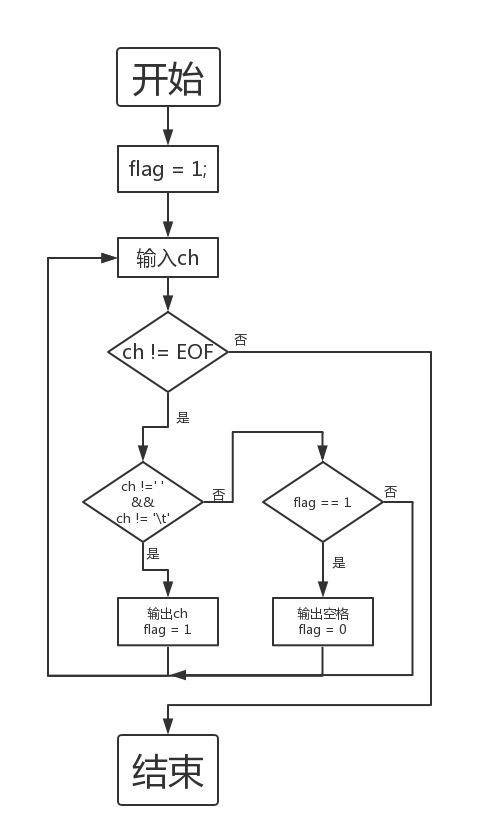


**图2-4 编程题1B的测试运行结果**

1. 编写一个程序,将输入的一行字符复制到输出，复制过程中将一个以上的空格字符用一个空格代替。

**解答：**

1. 算法流程如图2-5所示



**图2-5 编程题2的算法流程图**

1. 源程序清单

#include<stdio.h>

#include<string.h>

int main(void){

char ch;

int flag = 1;

while((ch = getchar()) != EOF){

if(ch != ' ' && ch != '\t'){

putchar(ch);

flag = 1;

}

else if(flag == 1){

putchar(' ');

flag = 0;

}

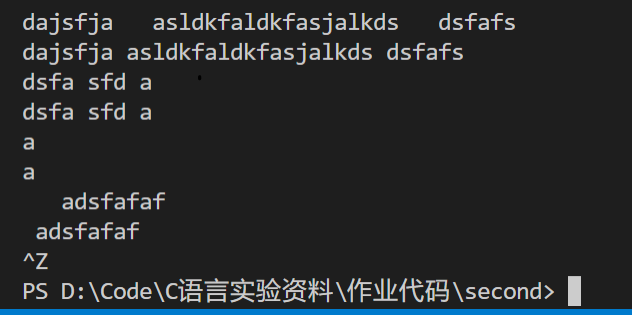
}

return 0;

}

3）测试

对应测试数据的运行结果截图



**图2-6 编程题2的测试用例1的运行结果**

1. 打印如下杨辉三角形。

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

每个数据值可以由组合计算（表示第i行第j列位置的值），而的计算如下：

 (i=0,1,2,…)

 (j=0,1,2,3,…,i)

本程序中为了打印出金字塔效果，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，程序编制过程中要注意区分。

**解答：**

1. 算法思路：

利用函数递归调用求出杨辉三角每个数据的值，循环输出，在printf函数中使用“%3d ”控制每一项数据间间隔空格数要求与金字塔效果

1. 源程序清单

#include<stdio.h>

int pascalT(int i, int j);

//该函数输出杨辉三角中的第i行第j个，从0开始计数

int main(void){

for(int i = 0, k = 18; i < 10; i++, k-=2){

for (int l = k; l >= 0; l--)

putchar(' ');

for (int j = 0; j < 10; j++){

if (pascalT(i, j))

printf("%3d ", pascalT(i, j));

else

printf(" ");

}

putchar('\n');

}

return 0;

}

int pascalT(int i, int j){

if(j == 0)

return 1;

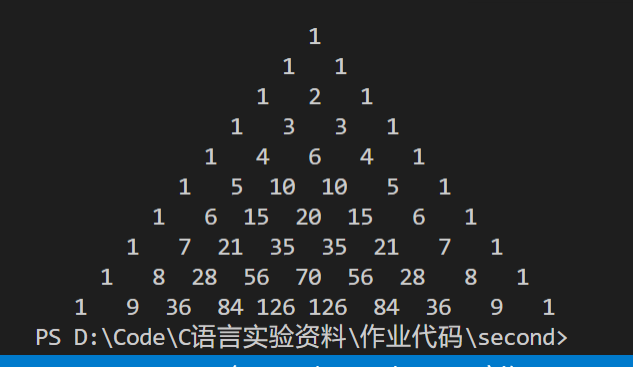
else

return pascalT(i, j-1)\*(i-j+1)/j;//使用递归

}

3)测试

对应测试数据的运行结果截图如图2-7所示

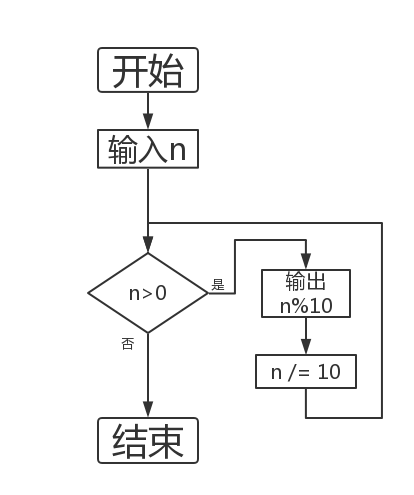


**图2-7 编程题3的测试的运行结果**

1. 编写一个程序，将用户输入的任意正整数逆转，例如，输入1234，输出4321。

**解答：**

1. 算法流程如图2-8所示



**图2-8 编程题4的程序流程图**

2) 源程序清单

#include<stdio.h>

int main(void){

int n;

scanf("%d", &n);

while(n > 0){

printf("%d", n % 10);

n /= 10;

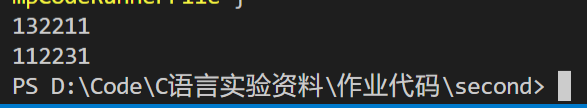
}

return 0;

}

3)测试

对应测试运行结果如图2-9所示。



**图2-9 编程题4的测试运行结果**

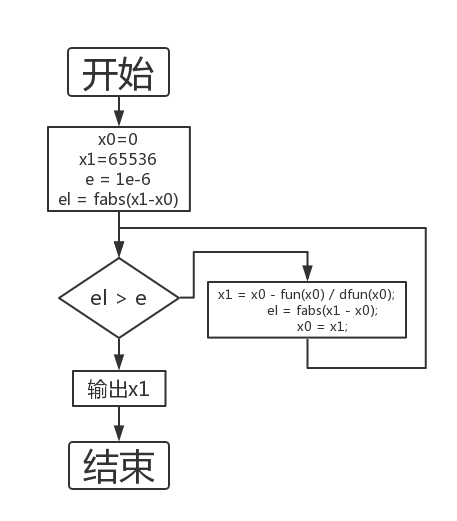
2.2.4. 选做题

编写并上机调试运行能实现以下功能的程序。

编写一个程序，用牛顿迭代法求方程满足精度e=10-6的一个近似根，并在屏幕上输出所求近似根。

**解答：**

1. 算法流程如图2-10所示



**图2-10 选做题的程序流程图**

1. 源程序清单

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

double fun(double x);

double dfun(double x);

int main(void){

double x0=0.0, x1= 65536.0;

double e = 1e-6;

double el;

el = fabs(x1 - x0);

while ( el > e){

x1 = x0 - fun(x0) / dfun(x0);

el = fabs(x1 - x0);

x0 = x1;

}

printf("%lf", x1);

return 0;

}

double fun(double x){

return 3 \* x \* x \* x - 4 \* x \* x - 5 \* x + 13;

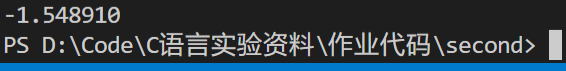
}

double dfun(double x){

return 4 \* x \* x - 8 \* x - 5;

}

3) 对应测试数据的运行结果截图如图2-11所示



**图2-11 选做题的测试的运行结果**

## 2.3 实验小结

第二次C语言上机实验，主要熟悉了复合语句、if语句、switch语句的使用，以及for、while、do-while三种基本的循环控制语句的使用，并了解了转移语句与标号语句。

1. **改错实验**

进一步提醒我，对于一些语法的细节需要格外注意，有时一个打字错误带来的bug需要耗费大量精力去找出。同时，需要对数据类型的范围及精度有认识和概念。同时，学习使用了for语句

1. **程序修改替换实验**

此题让我感受到用不同语句完成相同任务，以及已知一算法，对其求反的思想，同时与第一题改错结合，练习使用了三种循环语句，感受到了三种语句的利弊。

1. **程序设计实验**

工资税一题，了解到了税款的算法与switch语句的使用。处理空格的题目中，利用flag变量记录当前字符状态以决定是否需要输出单个空格，十分有效，可以迁移运用到其他程序设计中。杨辉三角题首先让我感受到了我国古代数学家的智慧，其次，再次感受到了递归的威力，将普通的迭代算法换成了简洁的递归写法。正整数逆转中，继续熟悉循环的利用。

1. **选做题**

此题首先了解了牛顿迭代法求解方程近似根的思想，正如其名，利用循环迭代求解是十分自然的思路，在实际编程中，使用到了库函数fabs()。