

**课 程 实 验 报 告**

**课程名称： C语言程序设计实验**

**专业班级：计算机类（二）2007班**

**学 号：U202017180**

**姓 名：吴彬杰**

**指导教师：万琳**

**报告日期：2020年10月28日**

**软件学院**

**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)

# 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

**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。

01 /\* 实验2-1改错题程序：合数判断器\*/

02 #include <stdio.h>

03 int main( )

04 {

05 int i, x, k, flag = 0;

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

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

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

09 if (!x%i) {

10 flag = 1;

11 break;

12 }

13 if(flag=1) printf("%d是合数", x);

14 else printf("%d不是合数", x);

15 }

16 return 0;

17 }

**解答：**

1. 错误修改：

1）flag初始化语句放入内层循环;

2）考虑x==2特殊情况;

完整任务代码如下:

/\* 实验2-1改错题程序：合数判断器\*/

#include <stdio.h>

#include <math.h>

int main(void)

{

    int i, x, k,flag;

    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 = floor(sqrt(x)+0.5); i <= k; i++)

            if ((x % i) == 0)

            {

                flag = 1;

                break;

            }

        if (x == 2)

        {

            printf("%d is a prime number\n", x);

            continue;

        }

        if (flag == 1)

            printf("%d is not a prime number\n", x);

        else

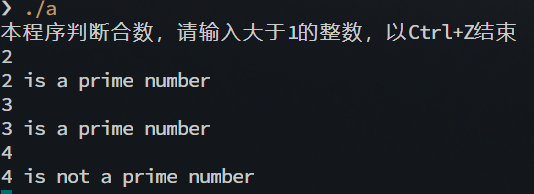
            printf("%d is a prime number\n", x);

    }

    return 0;

}

(2) 错误修改后运行结果



**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。

/\* 实验2-1改错题程序：合数判断器\*/

#include <stdio.h>

#include <math.h>

int isPrime(int a);

int main(void)

{

    int i, x, k,flag;

    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)

    {

        if (isPrime(x))

            printf("%d is a prime number\n", x);

        else

            printf("%d is not prime number\n", x);

    }

    return 0;

}

int isPrime(int a)

{

    if (a==2) return 114514;

    for (int i = 2, k = floor(sqrt(a)); i <= k; i++)

        if ((a % i) == 0)

            return 0;

    return 114514;

}

（2）修改实验2-1程序，将for循环改用do-while循环。

/\* 实验2-1改错题程序：合数判断器\*/

#include <stdio.h>

#include <math.h>

int main(void)

{

    int i, x, k, flag;

    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;

        i = 2;

        k = floor(sqrt(x));

        do

        {

            if ((x % i) == 0)

            {

                flag = 1;

                break;

            }

            i++;

        } while (i <= k);

        if (x == 2)

        {

            printf("%d is a prime number\n", x);

            continue;

        }

        if (flag == 1)

            printf("%d is not a prime number\n", x);

        else

            printf("%d is a prime number\n", x);

    }

    return 0;

}

（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。

/\* 实验2-1改错题程序：合数判断器\*/

#include <stdio.h>

#include <math.h>

int isPrime(int a);

int isPurely3unPrime(int a);

int main(void)

{

    for (int i=100;i<=999;i++)

        if (isPurely3unPrime(i))

            printf("%d\n",i);

    return 0;

}

int isPrime(int a)

{

    if (a == 2)

        return 1;

    for (int i = 2, k = floor(sqrt(a)); i <= k; i++)

        if ((a % i) == 0)

            return 0;

    return 1;

}

int isPurely3unPrime(int a)

{

    if ((!isPrime(a)) && (!isPrime(a/10)) && (!isPrime(a/100)))

        return 1;

    else

        return 0;

}

**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；x＞5000，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：

1. 算法流程图如图2-1所示:



2）程序清单

#include <stdio.h>

int main(void)

{

    int x;

    double tax = 0;

    scanf("%d", &x);

    if (x >= 1000 && x < 2000)

        tax = (x - 1000) \* 0.05;

    if (x >= 2000 && x < 3000)

        tax = 50 + (x - 2000) \* 0.1;

    if (x >= 3000 && x < 4000)

        tax = 150 + (x - 3000) \* 0.15;

    if (x >= 4000 && x <= 5000)

        tax = 300 + (x - 4000) \* 0.2;

    if (x > 5000)

        tax = 500 + (x - 5000) \* 0.25;

    printf("%.2lf", tax);

    return 0;

}

#include <stdio.h>

int main(void)

{

    int x, level;

    double tax = 0;

    scanf("%d", &x);

    level = x / 1000;

    switch (level)

    {

    case 0:

        break;

    case 1:

        tax = (x - 1000) \* 0.05;

        break;

    case 2:

        tax = 50 + (x - 2000) \* 0.1;

        break;

    case 3:

        tax = 150 + (x - 3000) \* 0.15;

        break;

    case 4:

        tax = 300 + (x - 4000) \* 0.2;

        break;

    default:

        tax = 500 + (x - 5000) \* 0.25;

        break;

    }

    printf("%.2lf", tax);

    return 0;

}

1. 测试数据

根据题意分别选取测试数据，如表2-1所示：

表2-1 编程题1的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 程序输入 | 理论结果 | 运行结果 |
| 用例1 | 5000 | 500 | 500 |
| 用例2 | 200 | 0 | 0 |
| 用例3 | 1000 | 0 | 0 |

（2）将输入的正文复制到输出，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)算法流程图如图2-2所示:



图2-2 编程题2的程序流程图

2) 程序清单：

#include <stdio.h>

int main(void)

{

    char ch[BUFSIZ];

    while (scanf("%s", ch))

    {

        fputs(ch, stdout);

        putchar(' ');

    }

    return 0;

}

3)测试数据：

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 程序输入 | 理论结果 | 运行结果 |
| 用例1 | aswecan | aswecan | aswecan |
| 用例2 | Fa q | Fa q | Fa q |
| 用例3 | 114 51 4 | 114 51 4 | 114 51 4 |

1. 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。

1. 算法流程图如图2-3所示:



图2-3 编程题3的程序流程图

1. 程序清单：

#include <stdio.h>

#include <string.h>

int main(void)

{

    int a[10][10];

    memset(a, 0, sizeof(a));

    for (int i = 0; i <= 9; i++)

    {

        a[i][0] = 1;

        a[i][i] = 1;

    }

    for (int i = 2; i <= 9; i++)

        for (int j = 1; j < i; j++)

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

    for (int i = 0; i <= 9; i++)

    {

        for (int j = 0; j <= 8 - i; j++)

            printf("  ");

        for (int j = 0; j <= i; j++)

            printf("%3d ", a[i][j]);

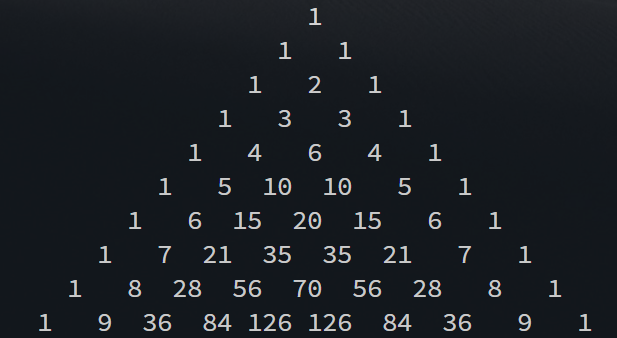
        printf("\n");

    }

    return 0;

}

1. 输出：



（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。

1）算法思路：

分离后三位，进行判断，合适则输出；

2）程序清单：

#include <stdio.h>

int main(void)

{

    int sum;

    for (int i = 100; i <= 999; i++)

    {

        sum = i \* i;

        if ((sum % 1000) == i)

            printf("%d\n", i);

    }

    return 0;

}

3)输出结果：

376

625

4. 选做题

（1）判断给定的字符串是否是合法的C整型常量，如果是，则输出Yes；不是，则输出No。例如，0xabL是C整形常量，而092不是C整形常量。要求程序能够循环接受用户的输入，每行输入一个字符串，给出判定结果，直至输入Ctrl+Z结束。

1）思路一：C标准库函数;

思路二：先判断进制，然后判断后缀，然后循环判断是否达标，全过程用一个flag进行标记；

2）程序清单：

#include <stdio.h>

int main(void)

{

    char a[BUFSIZ], b;

    int num;

    gets(a);

    if (sscanf(a, "%i%c", &num, &b) == 1)

        printf("Valid.");

    else

        printf("Invalid.");

    return 0;

}

#include <stdio.h>

#include <string.h>

int oct();

int dec();

int hex();

char str[1024];

int main(void)

{

    memset(str, 0, sizeof(str));

    while (gets(str))

    {

        toLower();

        printf("%s",str);

        if (str[0] < '0' || str[0] > '9')

        {

            printf("No");

            continue;

        }

        switch (str[0])

        {

        case '0':

            if (strlen(str) == 1)

                printf("Yes");

            else

            {

                if (str[1] == 'x')

                {

                    if (hex())

                        printf("Yes");

                    else

                        printf("No");

                }

                else if (oct())

                    printf("Yes.");

                else

                    printf("No");

                break;

            }

        default:

            if (dec())

                printf("Yes");

            else

                printf("No");

            break;

        }

    }

    return 0;

}

int oct()

{

    for (int i = 1; i < strlen(str); i++)

        if (str[i] < '0' || str[i] > '7')

            return 0;

    return 1;

}

int hex()

{

    for (int i = 2; i < strlen(str); i++)

        if (str[i] < '0' || str[i] > 'f' || (str[i] > '9' && str[i] < 'a'))

            return 0;

    return 1;

}

int dec()

{

    for (int i = 1; i < strlen(str); i++)

        if (str[i] < '0' || str[i] > '9')

            return 0;

    return 1;

}

3）测试数据如表2-4所示：

表2-4 编程题4的测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| 测试用例 | 程序输入 | 理论结果 | 运行结果 |
| 用例1 | 092 | Invalid | Invalid |
| 用例2 | 0x1f | Valid | Valid |
| 用例3 | 075 | Valid | Valid |

1. 输入正整数x（2⩽x⩽79），输出所有形如abcde/fghij=x的表达式，其中a～j由不同的数字0～9组成。例如：x=32时，输出为：75168/02349=32。

**解答：**

1. 算法思路：

枚举数据，拆分放入一个数组，用桶来判断是否重合，确认后输出。

1. 程序清单：

#include <stdio.h>

#include <math.h>

#include <string.h>

int abc[10];

int isABCDE();

void breakab(int a);

void breakfg(int a);

int main(void)

{

    int x;

    scanf("%d", &x);

    if (x < 2 || x > 79)

    {

        printf("Fuck you!");

        return 114514;

    }

    int mx = floor((98765 / (double)x));

    for (int ij = 1234; ij <= mx; ij++)

    {

        breakab(ij);

        breakfg(ij \* x);

        if (isABCDE())

            printf("%d%d%d%d%d/%d%d%d%d%d=%d", abc[0], abc[1], abc[2], abc[3], abc[4], abc[5], abc[6], abc[7], abc[8], abc[9], x);

    }

    return 0;

}

void breakab(int a)

{

    if (a <= 9999)

    {

        abc[5] = 0;

        abc[6] = a / 1000;

        abc[7] = a % 1000 / 100;

        abc[8] = a % 100 / 10;

        abc[9] = a % 10;

    }

    if (a >= 10000)

    {

        abc[5] = a / 10000;

        abc[6] = a % 10000 / 1000;

        abc[7] = a % 1000 / 100;

        abc[8] = a % 100 / 10;

        abc[9] = a % 10;

    }

}

void breakfg(int a)

{

    if (a <= 9999)

    {

        abc[0] = 0;

        abc[1] = a / 1000;

        abc[2] = a % 1000 / 100;

        abc[3] = a % 100 / 10;

        abc[4] = a % 10;

    }

    if (a >= 10000)

    {

        abc[0] = a / 10000;

        abc[1] = a % 10000 / 1000;

        abc[2] = a % 1000 / 100;

        abc[3] = a % 100 / 10;

        abc[4] = a % 10;

    }

}

int isABCDE()

{

    int sum = 0, t[10];

    memset(t, 0, sizeof(t));

    for (int jk = 0; jk <= 9; jk++)

        t[abc[jk]] = 1;

    for (int jk = 0; jk <= 9; jk++)

        sum += t[jk];

    if (sum == 10)

        return 1;

    else

        return 0;

}

1. 测试数据：

输入：32

理论输出：75168/02349=32

实际输出：75168/02349=32

## 1.3 实验小结

（1）编写程序经验来自于实践。将算法转变为真正的代码过程中会出现诸多的细节问题，源于实践，需要实践来解决。

（2）循环结构需要谨慎使用多层循环，循环量指数级增长，容易造成大量运算降低效率，应尽可能寻求算法优化。