



江苏理工学院
JIANGSU UNIVERSITY OF TECHNOLOGY

计算机工程学院

实验报告

课程名称：____程序设计基础____

实验地点：____60-____

专业班级：____21 软件 3R____ 学号：____2021144301____

学生姓名：____王大殿____

指导教师：____由从哲____

2021 年 月 日

学号	2021144301	姓名	王大殿	实验日期	2021.09.17
实验名称	熟悉 C 程序上机环境				
实验目的：1. 熟悉 Visual C 6.0 编程环境； 2. 编写简单的 C 语言程序。					
实验任务、步骤与结果： 任务 1：交换变量的值 题目描述：编写一个 C 程序，输入两个整数 a，b，交换 a 和 b 的数值，然后输出代码					
<pre>#include<stdio.h> void NumberChange(int *, int *); int main(int argc, char* argv[]) { int a, b; scanf("%d%d",&a,&b); NumberChange(&a,&b); printf("%d %d",a,b); } void NumberChange(int *a, int *b){ int i; //i 作为媒介，交换 a 和 b 的地址 i=*a; *a=*b; *b=i; }</pre>					

运行结果:

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_01
1 3
3 1
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_01
-1 1
1 -1
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$
```

任务 2: 方程求解

题目描述: 键盘输入 x 的值, 求方程 $y=x*x+2*x-10$ 所对应的 y 值。

```
#include<stdio.h>
```

```
int QuadraticEquation(int);
```

```
int main(int argc, char* argv[])
```

```
{
```

```
    int x,y=0;
```

```
    scanf("%d",&x);
```

```
    y=QuadraticEquation(x);
```

```
    printf("%d",y);
```

```
    return 0;
```

```
}
```

```
int QuadraticEquation(int x){
```

```
    int y=0;
```

```
    y=x*x+2*x-10;
```

```
return y;

}
```

运行结果:

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_02
0
-10
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_02
1
-7
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_02
10
110
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$
```

任务 3：华氏转换为摄氏

题目描述：有人用温度计测量出华氏法表示的温度(f)，进要求把它转换为以摄氏法表示的温度(c)

```
#include<stdio.h>
```

```
float FtoC(float);
```

```
int main(int argc, char* argv[])
```

```
{
```

```
    float F,C;
```

```
    scanf("%f",&F);
```

```
    C=FtoC(F);
```

```
    printf("%.2f\n",C);
```

```
    return 0;
```

```
}
```

```
float FtoC(float F){
    float C=0;

    C=5*(F-32)/9;  //必须是浮点数除

    return C;
}
```

运行结果:

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ gcc BaseOfCProgram_03.c -o BaseOfCProgram_03
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_03
32
0.00
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_03
40
4.44
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$
```

任务 4：计算存款利息

题目描述：计算存款利息。有 1000 元，想存一年。有 3 种方法可选：（1）活期，年利率为 r_1 ；（2）一年定期，年利率为 r_2 ；（3）存两次半年定期，年利率为 r_3 。请分别计算出一年后按 3 种方法所得到的本息和。

```
#include<stdio.h>
```

```
int main() {
```

```
    float Principal=1000;
```

```
    float r1,r2,r3;
```

```
    double p1,p2,p3;
```

```
    scanf("%f%f%f",&r1,&r2,&r3);
```

```
    p1=Principal+Principal*r1;
```

```
    p2=Principal+Principal*r2;
```

```
    p3=Principal+Principal*r3;
```

```
    printf("p1=%.6lf p2=%.6lf p3=%.6lf\n",p1,p2,p3);
```

```
return 0;
```

```
}
```

运行结果：

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ gcc BaseOfCProgram_04.c -o BaseOfCProgram_04
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$ ./BaseOfCProgram_04
0.0036 0.0225 0.0198
p1=1003.599976 p2=1022.500000 p3=1019.799988
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验1$
```

发现问题与分析：

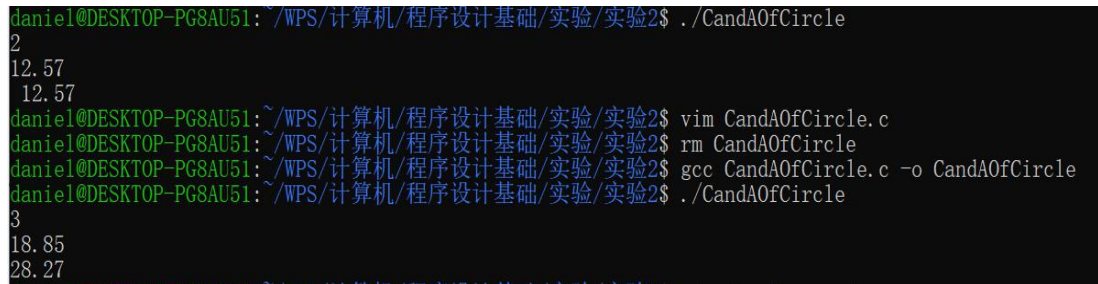
问题①：任务三中的温度转换，一开始没有设置好数据类型，导致算出来的结果为 0，将数值类型改为 float 过后，可以正常计算。

问题②：任务四中遇到了和任务三相似的问题，但数据类型已经是 float 类型了，经过排查，是因为 scanf 函数中变量前忘记写取地址符号&，修改过后程序正常运行。

备注：因 VC6++ 在 win10 上兼容性问题较多，编译采用了 win10 内置的 Ubuntu，等到学校可以在机房操作的时候再用 VC6++。

说明：

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	2021.09.28
实验名称	顺序程序设计				
<p>实验目的：1. 进一步熟悉 Visual C 6.0 编程环境；</p> <p>2. 掌握 C 语言变量与常量的使用；</p> <p>3. 编程简单的 C 语言程序。</p>					
<p>实验任务、步骤与结果：</p> <p>任务 1：求圆的周长与面积</p> <p>题目描述</p> <p>任务编制一个计算圆的面积与周长的程序，要求从键盘输入圆的半径，打印输出圆的周长与面积。</p> <p>代码：</p> <pre>#include<stdio.h> #define PI 3.14159 int main(){ float r, c, a; //radius, circumference, area scanf("%f",&r); c=r*PI*2; a=PI*r*r; printf("%.2f\n%.2f\n",c,a); return 0; }</pre> <p>运行结果</p>  <pre>daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2\$./CandAOfCircle 2 12.57 12.57 daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2\$ vim CandAOfCircle.c daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2\$ rm CandAOfCircle daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2\$ gcc CandAOfCircle.c -o CandAOfCircle daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2\$./CandAOfCircle 3 18.85 28.27</pre> <p>任务 2：大小写转换</p>					

题目描述

从键盘上任意输入一个字符，如果该字符为小写字母或大写字母，分别替换成对应的大写字母或小写字母，其它字符不变，如 a→A，B→b；\$→\$。

代码：

```
#include<stdio.h>

int main(){
    char c;
    c=getchar();
    if(c>=65 && c<=90){
        putchar(c+32);
    }else if(c>=97 && c<=122){
        putchar(c-32);
    }else{
        putchar(c);
    }
    putchar('\n');
    return 0;
}
```

运行结果：



```
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验2$ ./CaseConversion
e
E
```

任务 3：分解 9 位长整数

题目描述

编写程序，输入一个 9 位的长整数，将其分解为三个三位的基本整数并输出，其中个、十、百位为一个整数，千、万、十万位为一个整数，百万、千万、亿位为一个整数。例如 123456789 分解为 789、456 和 123。

代码：

```
#include<stdio.h>

int main(){
    int l, b1, b2, b3;
    scanf("%d",&l);
    b1=l%1000;
```



```

b2=(l/1000)%1000;
b3=l/1000000;
printf("%d %d %d\n",b1,b2,b3);
return 0;
}

```

运行结果：

```

daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2$ ./DecompositionOfIntegers
123456789
789 456 123

```

任务 4：是否被 3 和 5 整除

题目描述

从键盘上输入任一正整数（如果输入不符合规范，打印 “Incorrect input!”，退出程序），判断该整数能否同时被 3 和 5 整除，能则在屏幕上输出 3 和 5，否则输出该数的相反数。

代码：

```

#include<stdio.h>

int main(){
    int i;
    if(scanf("%d",&i) && i!=0){
        if(!(i%3 && i%5)){
            printf("3 5\n");
        }else{
            printf("%d\n",-i);
        }
    }else{
        printf("Incorrect input\n");
    }
    return 0;
}

```

运行结果：

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2$ ./DivisibelBy3And5
0
Incorrect input
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2$ ./DivisibelBy3And5
15
3 5
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2$ ./DivisibelBy3And5
-15
3 5
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验2$
```

发现问题与分析:

任务 2 的判断条件一开始写的是 “if(c>=A && c<=Z)” 和 “if(c>=a && c<=z)”，在编译的时候报错，报错内容如下：

```
CaseConversion.c: In function 'main' :
CaseConversion.c:8:8: error: 'A' undeclared (first use in this function)
 8 | if(c>=A && c<=Z) {
    |         ^
CaseConversion.c:8:8: note: each undeclared identifier is reported only once for each function it appears in
CaseConversion.c:8:16: error: 'Z' undeclared (first use in this function)
 8 | if(c>=A && c<=Z) {
    |                ^
CaseConversion.c:10:14: error: 'a' undeclared (first use in this function)
10 | }else if(c>=a && c<=z) {
    |              ^
CaseConversion.c:10:22: error: 'z' undeclared (first use in this function)
10 | }else if(c>=a && c<=z) {
    |                      ^
```

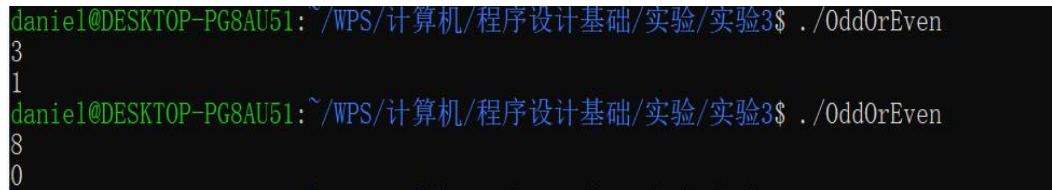
c 语言中不可以直接把字母比较大小，自己有点想当然的把写 JS 时的一些习惯带过来了。后面将判断条件改为与相应的 ASCII 码比较。

任务 4 在编写的时候直接用 if(scanf(“%d”,&i))来判断输入是否合规，忽略了输入值为 0 的情况，后面按照样例进行输入输出的时候发现了这个问题并补上了输入为 0 时报错这一条件。

备注:

说明:

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	2021.10.8
实验名称	选择结构程序设计				
实验目的： 1. 掌握条件表示式； 2. 学会正确使用逻辑运算符和逻辑表达式； 3. 熟练使用 if 语句和 switch 语句求解问题。					
实验任务、步骤与结果： 任务 1：判断奇偶数 题目描述 输入一个整数 n，判断是奇数还是偶数，并输出判断结果。（如果是奇数则输出 1，如果是偶数则输出 0）(1<=n<=1000) 代码： #include<stdio.h> int main(){ int n; scanf("%d",&n); printf("%d\n",n%2); return 0; } 运行结果  任务 2：成绩转换 题目描述： 给出一百分制成绩，要求输出成绩等级 A、B、C、D、E。90 分以上为 A，80~89 为 B，70~79 分为 C，60~69 分为 D，60 分以下为 F。 代码： #include<stdio.h> int main(){ int score, s_t; char grade; printf("Input a student score:\n"); scanf("%d",&score); s_t=score/10;					

```

    if(score<0 || score>100){
        printf("Input error!\n");
    }else{
        switch(s_t){
            case 10:
            case 9:
                grade='A';
                break;
            case 8:
                grade='B';
                break;
            case 7:
                grade='C';
                break;
            case 6:
                grade='D';
                break;
            default:
                grade='E';
        }
        printf("the student grade:\n%c\n",grade);
    }
    return 0;
}

```

运行结果：

```

daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./ScoreToGrade
Input a student score:
110
Input error!
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./ScoreToGrade
Input a student score:
100
the student grade:
A
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./ScoreToGrade
Input a student score:
40
the student grade:
E
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$

```

任务 3：企业发放的奖金根据利润提成

题目描述：

利润(I)低于或等于 10 万元时，奖金可提 10%；

利润高于 10 万元，低于 20 万元时，低于 10 万元的部分按 10%提成，高于 10 万元的部分，可提成 7.5%；

20 万到 40 万之间时，高于 20 万元的部分，可提成 5%；

40 万到 60 万之间时高于 40 万元的部分，可提成 3%；

60 万到 100 万之间时，高于 60 万元的部分，可提成 1.5%；

高于 100 万元时，超过 100 万元的部分按 1%提成，从键盘输入当月利润 I，求应发放奖金总数（精确到小数点后面两位）？

提示：可以画一根数轴，把每段利润的提成数标记出来，再编程实现。

代码：

```
#include<stdio.h>
int main(){
    float i, bonus;
    printf("Input the profit for the month:\n");
    scanf("%f",&i);
    if(i<0){
        printf("No Bonus!\n");
    }else{
        if(i<=100000){
            bonus=i*0.1;
        }else if(i>100000 && i<=200000){
            bonus=100000*0.1+(i-100000)*0.075;
        }else if(i>200000 && i<=400000){
            bonus=100000*0.1+(200000-100000)*0.075+(i-200000)*0.05;
        }else if(i>400000 && i<=600000){
            bonus=100000*0.1+(200000-100000)*0.075+(400000-200000)*0.05+(i-400000)*0.03;
        }else if(i>600000 && i<=1000000){
            bonus=100000*0.1+100000*0.075+200000*0.05+200000*0.03+(i-600000)*0.015;
        }else{
            bonus=100000*0.1+100000*0.075+200000*0.05+200000*0.03+400000*0.015+(i-1000000)*0.01;
        }
        printf("The bonus is %.2f.\n",bonus);
    }
    return 0;
}
```

运行结果：

```

daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$ ./Bonus_calculate
Input the profit for the month:
10000
The bonus is 1000.00.
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$ ./Bonus_calculate
Input the profit for the month:
150000
The bonus is 13750.00.
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$ ./Bonus_calculate
Input the profit for the month:
250000
The bonus is 20000.00.
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$ ./Bonus_calculate
Input the profit for the month:
450000
The bonus is 29000.00.
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$ ./Bonus_calculate
Input the profit for the month:
-1000
No Bonus!
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$

```

任务 4：水仙花数

题目描述：

水仙花数是指这样一种三位数，各个数位的立方和加起来等于这个数本身，如 $153=1^3+5^3+3^3$ ，试编制一个程序，验证从键盘上输入的一个数是否为水仙花数。

如果输入数据不正确（不是三位正整数），则提示：

Error input!。

如果输入整数为水仙花数则按下列格式输出：

$153=1^3+5^3+3^3$

否则输出：

Not a narcissistic number

代码：

```

#include<stdio.h>
int IsNarcissisticNumber(int);
int main(){
    int n;
    printf("Input a number(100<=n<1000):\n");
    scanf("%d",&n);
    int h=n/100;
    int t=(n-h*100)/10;
    int s=n%10;
    if(n<100 || n>=1000){
        printf("Error input!\n");
    }else if(IsNarcissisticNumber(n)){
        printf("%d=%d*%d*%d+%d*%d*%d+%d*%d*%d\n",n,h,h,h,t,t,t,s,s,s);
    }else{
        printf("Not a narcissistic number!\n");
    }
    return 0;
}

```

```

    }

    int IsNarcissisticNumber(int n){
        int h=n/100;
        int s=n%10;
        int t=(n-h*100)/10;
        if(n==h*h*h+t*t*t+s*s*s){
            return 1;
        }else{
            return 0;
        }
    }
}

```

运行结果:

```

daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./Narcissistic_Number
Input a number(100<=n<1000):
3
Error input!
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./Narcissistic_Number
Input a number(100<=n<1000):
153
153=1*1*1+5*5*5+3*3*3
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./Narcissistic_Number
Input a number(100<=n<1000):
998
Not a narcissistic number!
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$

```

任务 5: 一年的第几天

题目描述:

输入某年某月某日, 判断这一天是这一年的第几天?

提示: 以 3 月 5 日为例, 应该先把前两个月的天数加起来, 然后再加上 5 天即本年的第几天, 特殊情况, 闰年且输入月份大于 3 时需考虑多加一天。

代码:

```

//输入某年某月, 判断这一天是这一年的第几天
#define bool int
#define true 1
#define false 0
#include<stdio.h>
bool IsLeapYear(int);
int main(){
    int year, month, day;
    int date=0;
    printf("Please input the date(year-month-day,like 2015-5-3):\n");
    scanf("%d-%d-%d",&year,&month,&day);
    switch(month){
        case 1:

```

```
        date=day;
        break;
    case 2:
        date=31+day;
        break;
    case 3:
        date=59+day;
        break;
    case 4:
        date=90+day;
        break;
    case 5:
        date=120+day;
        break;
    case 6:
        date=151+day;
        break;
    case 7:
        date=181+day;
        break;
    case 8:
        date=212+day;
        break;
    case 9:
        date=243+day;
        break;
    case 10:
        date=273+day;
        break;
    case 11:
        date=304+day;
        break;
    case 12:
        date=334+day;
        break;
    default:
        date=-1;
        break;
}
if(date==-1){
    printf("Error!\n");
}else if(IsLeapYear(year) && month>2){
    date=date+1;
```



```

        printf("%d\n",date);
    }else{
        printf("%d\n",date);
    }
    return 0;
}

bool IsLeapYear(int year){
    if(year%4==0 && year%100!=0 || year%400==0){
        return true;
    }else{
        return false;
    }
}

```

运行结果:

```

daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./DayOfYear
Please input the date(year-month-day, like 2015-5-3):
1900-12-1
335
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./DayOfYear
Please input the date(year-month-day, like 2015-5-3):
1996-4-1
92
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$ ./DayOfYear
Please input the date(year-month-day, like 2015-5-3):
2016-5-30
151
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验3$

```

任务 6: 谁是杀手

题目描述:

日本某地发生了一件谋杀案, 警察通过确定杀人凶手必为 4 个嫌疑犯中的一个。以下为 4 个嫌疑犯的会供词。

A 说: 不是我。

B 说: 是 C。

C 说: 是 D。

D 说: C 在胡说。

已知 3 个人说了真话, 1 个人说的是假话。现在请根据这些信息, 写一个程序来确定到谁是凶手。

代码:

```

#include<stdio.h>
int main(){
    char killer;
    int result,r1,r2,r3,r4;
    for(killer='A';killer<='D';killer++){

```

```
    r1=killer!='A';
    r2=killer=='C';
    r3=killer=='D';
    r4=killer!='D';
    result=r1+r2+r3+r4;
    if(result==3){
        printf("Killer is %c.\n",killer);
        break;
    }
}
return 0;
}
```

运行结果:



```
daniel@DESKTOP-PG8AU51: ~/WPS/计算机/程序设计基础/实验/实验3$ ./WhoIsKiller
Killer is C.
```

发现问题与分析:

1. 在任务 5 中, 需要编写一个判断闰年的程序。我将函数的返回值类型设置为 bool 值, 不过在编译的时候报错了, 显示类型未定义。经过搜索后发现, C 默认没有布尔值, 判断条件是靠返回 0 或 1 来判断真假。所以后面我将 bool 定义为 int 类型, true 定义为 1, false 定义为 0。

备注:

说明:

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小, 但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称	循环结构程序设计				
<p>实验目的：1. 掌握循环结构基本思想与语法；</p> <p>2. 利用循环结构解决一般问题；</p> <p>3. 循环程序设计编码规范。</p>					
实验任务、步骤与结果：					
任务 1：寻找水仙花数					
题目描述					
<p>水仙花数是指这样一种三位数，各个数位的立方和加起来等于这个数本身，如 $153=1^3+5^3+3^3$，试编制一个程序，验证从键盘上输入的一个数是否为水仙花数。</p> <p>输出所有的水仙花数，每行 1 个。</p>					
代码：					
<pre>#include<stdio.h> int IsNarcissistic(int); int main(){ printf("以下为所有水仙花数： \n"); for(int i=100; i<1000; i++){ if(IsNarcissistic(i)==1){ printf("%d\n",i); } } return 0; } int IsNarcissistic(int n){ if(n<100 n>=1000){ return -1; //输入错误 } int s=n%10; //个位 int h=n/100; //百位 int t=(n-h*100)/10; //十位 if(s*s*s+h*h*h+t*t*t==n){ return 1; //是水仙花数 }else{ return 0; //不是水仙花数 } }</pre>					

运行结果

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./NarcissisticNumber
以下为所有水仙花数：
153
370
371
407
```

任务 2：九九乘法表

题目描述

修改下列程序的打印输出语句，打印输出如下图的乘法口诀表。

```
#include "stdio.h"
```

```
int main()
```

```
{
    float i,j;
    for(i=1;i<10;i++)
    {
        for(j=1;j<i+1;j++)
            printf("%f%f%f",i,j,i*j);
        putchar('\n');
    }
}
```

代码：

```
#include "stdio.h"
```

```
int main()
```

```
{
    int i,j;
    for(i=1;i<10;i++)
    {
        for(j=1;j<i+1;j++)
            printf("%d*%d=%d\t",i,j,i*j);
        putchar('\n');
    }
}
```

运行结果：

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./乘法表
1*1=1
2*1=2   2*2=4
3*1=3   3*2=6   3*3=9
4*1=4   4*2=8   4*3=12  4*4=16
5*1=5   5*2=10  5*3=15  5*4=20  5*5=25
6*1=6   6*2=12  6*3=18  6*4=24  6*5=30  6*6=36
7*1=7   7*2=14  7*3=21  7*4=28  7*5=35  7*6=42  7*7=49
8*1=8   8*2=16  8*3=24  8*4=32  8*5=40  8*6=48  8*7=56  8*8=64
9*1=9   9*2=18  9*3=27  9*4=36  9*5=45  9*6=54  9*7=63  9*8=72  9*9=81
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$
```

题目描述

修改完善程序输出如下图案：

```
#include"stdio.h"
int main()
{
    char c[7]={'a','b','c','d','e','f','g'};
    int i,j;
    for(i=0;i<7;i++)
    {
        for(j=7;j>i+1;j--)
            putchar(' ');
        for(j=0;j<2*i+1;j++)
            putchar(c[i]);
        putchar('\n');
    }

    for(_____)
    {
        for(_____)
            _____;
        for(j=0;j<2*i+1;j++)
            putchar(c[i]);
        _____;
    }
    return 0;
}
```

提示：可以使用数组记录字母，如 `char[7]={'a','b','c','d','e','f','g'}`；然后使用循环控制输出字母和空格个数，如输出字母 `a`，可使用语句 `printf("%c",a[0])`；

代码：

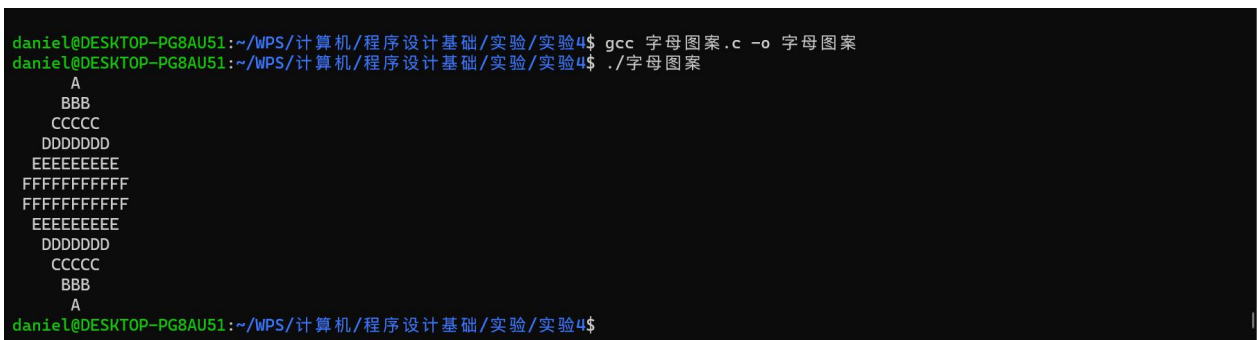
```
#include"stdio.h"
int main()
{
    char c[7]={'A','B','C','D','E','F','G'};
    int i,j;
    for(i=0;i<6;i++)
    {
        for(j=7;j>i+1;j--)
            putchar(' ');
        for(j=0;j<2*i+1;j++)
            putchar(c[i]);
        putchar('\n');
    }
}
```

```

    for(i=5; i>=0; i--)
    {
        for(j=5-i; j>=0; j--)
            putchar(' ');
        for(j=0; j<2*i+1; j++)
            putchar(c[i]);
        putchar('\n');
    }
    return 0;
}

```

运行结果：



```

daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ gcc 字母图案.c -o 字母图案
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./字母图案
  A
 BBB
CCCCC
DDDDDDD
EEEEEEEE
FFFFFFFFFFFF
FFFFFFFFFFFF
EEEEEEEE
DDDDDDD
CCCCC
 BBB
  A
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$

```

任务 5：求 10 个数的最小值和最大值

题目描述

输入 10 个整数，输出其最小值和最大值。

代码：

```

#include<stdio.h>
int input();
int main(){
    //输入部分
    int a[100];
    int count=0;
    //输入提示
    printf("请输入 10 个数，用任意字符隔开：\n");
    //接收返回的数组长度
    count=input(a);
    //根据返回长度建立数组
    int num[count];
}

```

```

//把 a 数组的值，赋值到 num 中
for(int i=0; i<count; i++){
    num[i]=a[i];
    //printf("%d\n",num[i]);
}

//比较最大最小值
int min=num[0], max=num[0];
for(int j=0; j<count; j++){
    if(num[j]>max){
        max=num[j];
    }else if (num[j]<min)
    {
        min=num[j];
    }else{
        continue;
    }
}
printf("%d\t%d\n",max, min);
return 0;
}
//用于输入一行数组，数组个数不超过 100，返回个数
int input(int *a){
    int len=0;
    char ch;
    //读取输入数组，直到读入回车字符则停止
    do{
        scanf("%d%c",&a[len],&ch);
        len++;
    }while(ch!='\n');
    return len;
}

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./最大最小数
请输入10个数，用任意字符隔开:
1 2 3 4 5 6 7 8 9 10
1      10
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ 

```

任务 6: 简单加密

题目描述

编程实现一个简单的加密方法，把所有字母用它后面第四个字母代替，数字和符号不变，如 A 替换成 E，z 替换成 d。如下图所示：



编写程序，从键盘上输入一串原文，按回车后输出密文，输入字符“#”退出程序。提示，可以使用 `getchar()` 从键盘上接收一个字符，用 `putchar()` 把字符输出到屏幕上。使用循环语句即可实现一次输入和输出多个字符。代码如下：

```
# include<stdio.h>
```

```
int main()
```

```
{ char c;
```

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

```
{
```

```
/*添加字符转换代码*/
```

```
    putchar(c);
```

```
}
```

```
}
```

代码：

```
/*
```

编程实现一个简单的加密方法，把所有字母用它后面第四个字母代替，数字和符号不变，如 A 替换成 E，z 替换成 d。

```
*/
```

```
#include<stdio.h>
```

```
int main(){
```

```
    char c;
```

```
    //输入字符“#”退出程序
```

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

```
{
```

```
    if((int)c>=65 && (int)c<=90){
```

```
        c=(c+4-65)%26+65;
```

```
    }else if ((int)c>=97 && (int)c<=122)
```

```
{
```

```

        c=(c+4-97)%26+97;
    }
    putchar(c);
}
putchar('\n');
return 0;
}

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ gcc 原文加密.c -o 原文加密
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./原文加密
abcxyz#
efgbcd
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./原文加密
ABCXYZ#
EFGBCD
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./原文加密
aAzZ#
eEdD
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ 

```

任务 7：斐波那契数列（Fibonacci sequence）

题目描述

斐波那契数列（Fibonacci sequence），又称黄金分割数列、因数学家列昂纳多·斐波那契（Leonardoda Fibonacci）以兔子繁殖为例子而引入，故又称为“兔子数列”，指的是这样一个数列：1、1、2、3、5、8、13、21、34、.....在数学上，斐波纳契数列以如下被以递归的方法定义： $F(1)=1$ ， $F(n)=F(n-1)+F(n-2)$ ($n \geq 2$, $n \in N^*$) 在现代物理、准晶体结构、化学等领域，斐波纳契数列都有直接的应用，为此，美国数学会从 1963 起出版了以《斐波纳契数列季刊》为名的一份数学杂志，用于专门刊载这方面的研究成果。

1	1	2	3
5	8	13	21
34	55	89	144
233	377	610	987
1597	2584	4181	6765
10946	17711	28657	46368
75025	121393	196418	317811
514229	832040	1346269	2178309
3524578	5702887	9227465	14930352
24157817	39088169	63245986	102334155
165580141	267914296	433494437	701408733
1134903170	1836311903	2971215073	4807526976
7778742049	12586269025	20365011074	32951280099
53316291173	86267571272	139583862445	225851433717
365435296162	591286729879	956722026041	1548008755920
2504730781961	4052739537881	6557470319842	10610209857723
17167680177565	27777890035288	44945570212853	72723460248141
117669030460994	190392490709135	308061521170129	498454011879264
806515533049393	1304969544928657	2111485077978050	3416454622906707
5527939700884757	8944394323791464	14472334024676220	23416728348467684
37889062373143904	61305790721611584	99194853094755488	160500643816367070
259695496911122560	420196140727489660	679891637638612220	1100087778366101900
1779979416004714000	2880067194370816000	4660046610375530500	7540113804746346500
12200160415121877000	19740274219868226000	31940434634990100000	51680708854858326000

要求：打印输出 4 年的兔子数，每 4 个月一行，每列占位宽度为 20。

提示：可使用更长的数据类型，防止整数溢出。

代码：

```
#include<stdio.h>
long long int Fibonacci(int);
int main(){
    int n;
    printf("In put the number of month:\n");
    scanf("%d",&n);
    long long int F;
    for(int i=1; i<=n; i++){
        F=Fibonacci(i);
        printf("%20lld",F);
        if(i%4==0){
            printf("\n");
        }
    }
    return 0;
}

long long int Fibonacci(int n){
    if(n==1){
        return 1;
    }else if (n==2)
    {
        return 1;
    }else{
        return Fibonacci(n-1)+Fibonacci(n-2);
    }
}
```

运行结果：

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$ ./Fibonacci_sequence
In put the number of month:
48
          1          1          2          3
          5          8          13         21
          34         55          89         144
          233        377         610         987
          1597       2584        4181        6765
          10946      17711       28657       46368
          75025      121393      196418      317811
          514229     832040     1346269     2178309
          3524578    5702887    9227465    14930352
          24157817   39088169   63245986   102334155
          165580141  267914296  433494437  701408733
          1134903170 1836311903 2971215073 4807526976
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验4$
```

发现问题与分析：在最大最小数那一题中，我一开始想不定义数组，只定义一个指针，写一个 input 函数，参数就是主函数中的指针。Input 函数用一个 do...while()函数作为主体，函数体为 “scanf("%d%c",a,&ch); a++;” 检测到'\n' 则停止。但实际运行时却出现了 segmentation fault。经查找发现该错误为存储器段错误，它会出现当程序企图访问 CPU 无法定址的存储器区块时。我猜想可能时指针出了问题，查询后发现时 a 这个指针在主函数定义过后没有初始化，是一个“野指针”。为了解决这个问题，我就没在主函数中再定义指针而是直接定义了一个数组，把数组赋给 input 函数。

备注：

说明：

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称	一维数组				
实验目的：					
1. 掌握一维数组的定义、赋值和输入输出的方法；					
2. 掌握与数组有关的算法（例如排序算法）。					
实验任务、步骤与结果：					
任务 1：数组插入数据					
题目描述					
有一个已排序的数组{2,5,7,10,14,17,20}，现输入一个整数，要求按原来排序的规律将它插入数组中。					
代码：					
<pre>#include<stdio.h> int main(){ int a[10]={2,5,7,10,14,17,20}; int n, i; scanf("%d",&n); for(i=0; i<7;i++){ if(n<=a[i]){ for(int j=7; j>i; j--){ a[j]=a[j-1]; //printf("%d\n",a[j]); } a[i]=n; //printf("%d\n",a[i]); break; }else if (n>a[6]) { a[7]=n; } } for(int k=0; k<8; k++){ printf("%d\t",a[k]); } printf("\n"); return 0; }</pre>					

运行结果:

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./数组插入
4
2      4      5      7      10     14     17     20
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./数组插入
31
2      5      7      10     14     17     20     31
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ █
```

任务 2: 数组逆序存储

题目描述:

将一个数组中的值按逆序重新存放。例如，原来顺序为 8，6，5，4，1。逆序后的顺序为 1，4，5，6，8。

代码:

```
#include<stdio.h>
int main(){
    int a[100];
    char ch;
    int len=0;
    do{
        scanf("%d%c",&a[len],&ch);
        len++;
    }while(ch!='\n');
    int a_mirror[len];
    for(int i=0; i<len; i++){
        a_mirror[i]=a[len-i-1];
        printf("%d ", a_mirror[i]);
    }
    printf("\n");
    return 0;
}
```

运行结果:

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ gcc 数组逆序输出.c -o 数组逆序输出
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./数组逆序输出
1 2 3 4 5 6
6 5 4 3 2 1
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./数组逆序输出
8 6 5 4 1
1 4 5 6 8
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ █
```

任务 3: 约瑟夫问题

题目描述

又称猴子选大王问题。 m 个猴子围成一圈，从 1 开始报数，报到 n 的猴子出局，剩下的猴子再从 1 开始报数，报到 n 者出局。最后剩下的猴子为大王。

编程输出最后留下的大王？

输入

1 行：

两个整数，猴子数 m 和报数 n

输出

大王的位置，如 King is a[1]=1

代码：

```
#include<stdio.h>
```

```
int main(){
```

```
    int m, n;
```

```
    printf("请输入猴子个数 m， 以及 n\n");
```

```
    scanf("%d%d",&m, &n);
```

```
    int len=m, i=1, k=0, count=0;
```

```
    //i 用来给循环计数，k 为数组下标，count 用于在循环中记录 0
```

```
    int a[len];
```

```
    for(int j=0; j<len; j++){
```

```
        a[j]=j+1;
```

```
    }
```

```
    while (len>1){
```

```
        k=(i+count-1)%m;
```

```
        if(a[k]!=0){
```

```
            if(i%n==0){
```

```
                a[k]=0;
```

```
                len--;
```

```
            }
```

```
            i++;
```

```
        }else{
```

```
            count++;
```

```
            //当 a[k]==0 时，count 加一，用于跳向下一位
```

```
        }
```

```
    }
```

```
    for(int i=0; i<m; i++){
```

```
        if(a[i]!=0){
```

```
            printf("King is a[%d]=%d\n",a[i],a[i]);
```

```
        }
```

```
    }
```

```
    return 0;
```

```
}
```

运行结果：

```

daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ gcc -g JosefProblem.c -o JosefProblem
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./JosefProblem
请输入猴子个数m, 以及n
8 5
King is a[3]=3
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./JosefProblem
请输入猴子个数m, 以及n
10 6
King is a[3]=3
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./JosefProblem
请输入猴子个数m, 以及n
20 2
King is a[9]=9
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ 

```

任务 4: 折半查找

题目描述

有 15 个数按由小到大顺序存放在一个数组中, 输入一个数, 要求用折半查找法找出该数是数组中第几个元素的值。如果该数不在数组中, 则打印出 “None”。

代码:

```

#include<stdio.h>
int binarySearch(int arr[], int, int, int);
int main(){
    int len=15, index=0, n=0;
    printf("Input the number you want:");
    scanf("%d",&n);
    int a[len];
    for(int i=0; i<len; i++){
        a[i]=i+1;
    }
    index=binarySearch(a, 0, len, n);
    if(index!=-1){
        printf("The index of %d is %d.\n",n,index);
    }else{
        printf("None!\n");
    }
}
int binarySearch(int arr[], int low, int high, int x){
    int mid=(low+high)/2;
    if(x>arr[high-1] || x<arr[low]){
        return -1;
    }
    if(arr[mid]==x){
        return mid;
    }else if(x<arr[mid]){
        return binarySearch(arr, low, mid, x);
    }else if(x>arr[mid]){
        return binarySearch(arr, mid, high, x);
    }
}

```


运行结果

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./Binary_Search
Input the number you want:1
The index of 1 is 0.
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./Binary_Search
Input the number you want:5
The index of 5 is 4.
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./Binary_Search
Input the number you want:-1
None!
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./Binary_Search
Input the number you want:19
None!
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$
```

任务 5：看到几座山

题目描述

众所周知，黄山是一片山（而不是一座山）。我们假设这些山排成了一排。每座山有各自的高度。现在游客们从最左边看山，有些山因为高度没有它左边的某座山高，就会被遮住，游客们就无法看到，也就是说对于一座山而言，只有当这座山比它左边所有的山都高的时候，游客们才能看到这座山。现在想请你告诉游客，他能看到几座山。

这个问题可以描述为找出数组中比前面所有元素都要大的元素个数。

代码：

```
#include<stdio.h>
int main(){
    int N=0;
    printf("Input the number of mountains: ");
    scanf("%d",&N);
    long int arr[N];
    char ch;
    int i=0, top=0, count=1;
    do{
        scanf("%ld%c",&arr[i],&ch);
        i++;
    }while(ch!='\n');
    for(int j=0; j<N-1; j++){
        if(arr[j]>top){
            top=arr[j];
        }
        if(arr[j+1]>top){
            count++;
        }
    }
    printf("You can see %d mountains.\n",count);
    return 0;
}
```

运行结果：

```
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ gcc 看到几座山.c -o 看到几座山
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./看到几座山
Input the number of mountains: 10
1 2 3 4 5 1 2 3 4 5
You can see 5 mountains.
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./看到几座山
Input the number of mountains: 5
5 4 3 2 1
You can see 1 mountains.
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$ ./看到几座山
Input the number of mountains: 5
3 4 1 8 5
You can see 3 mountains.
daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/实验/实验5$
```

发现问题与分析：

备注：

说明：

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称					
实验目的：					
<p>实验任务、步骤与结果：</p> <p>矩阵对角线之和</p> <p>题目描述</p> <p>求一个 3×3 矩阵左右对角线元素之和（去掉对角一交叉重复数）。</p> <p>输入</p> <p>3 行：</p> <p>一个 3 行 3 列的矩阵</p> <p>输出</p> <p>1 行：</p> <p>1 个整数，左右对角线之和</p> <p>代码：</p> <pre>#include<stdio.h> int main(){ int arr[3][3]; int sum=0; char ch; //二维数组输入，元素用空格隔开，输完一行按 enter for(int i=0; i<3; i++){ int len=0; do{ scanf("%d%c",&arr[i][len],&ch); len++; }while(ch!="\n"); } for(int j=0; j<3; j++){ //对角相加 sum+=arr[j][j]+arr[2-j][j]; } sum=sum-arr[1][1]; printf("%d\n",sum); return 0; }</pre> <p>运行结果</p>					

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./对角线之和
1 2 3
4 5 6
7 8 9
25
```

插入单词

题目描述

现有一组已经排好序的单词，根据字母顺序，将一个新单词插入队列中。

输入

若干行：

第 1 行，整数 n ，表示接下来输入 n 个单词

第 2~ $n+1$ 行， n 个从小到大排列的单词

第 $n+1$ 行，要插入的单词

输出

若干行：

插入后的单词，每行 1 个

```
#include<stdio.h>
#include<string.h>
int main(){
    int n;
    printf("Input the number of words: ");
    scanf("%d",&n);
    char arr[n+1][20];
    for (int i = 0; i <= n; i++)
    {
        scanf("%s",arr[i]);
    }
    for(int j=0; j<n; j++){
        if(strcmp(arr[n], arr[j])<0){
            //如果最后一个单词比第 j 个单词小，则第 j 个后面的单词向后移一位
            char str_t[20];
            strcpy(str_t,arr[n]);
            for(int k=n-1; k>=j; k--){
                strcpy(arr[k+1],arr[k]);
            }
            //把最后一个单词复制给第 j 个
            strcpy(arr[j],str_t);
            break;
        }
    }
    for (int i = 0; i <=n; i++)
    {
        printf("%s\n",arr[i]);
    }
}
```

```

    }

    return 0;
}

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./WordInsert
Input the number of words: 5
An
apple
bee
bike
meat
frog
An
apple
bee
bike
frog
meat

```

最小等待时间

题目描述

银行一共 n 个客户在排队，每个客户因业务不同，需要的时间从 $1 \sim m$ 分钟不等。银行大堂经理根据每个客户的业务类型，安排业务排队，使 n 个客户的总等待时间最少。例如，某个客户办理业务需要 5 分钟，他面前有 4 个客户在等待，则该客户产生的等待时间是 $5 \times 4 = 20$ 分钟，而如果 he 只需要 1 分钟，则 he 产生的等待时间是 4 分钟。

输入

2 行:

第 1 行, 1 个整数 n , 表示有 n 个客户

第 2 行, n 个整数, 表示每个客户需要等待时间

输出

1 行:

整数 n , 表示最少需要等待的时间

代码:

```

#include<stdio.h>
void BubbleSort(int a[], int n);
int main(){
    int n;
    printf("Enter the number of customers.\n");
    scanf("%d",&n);
    int a[n];
    int len=0, sum=0;
    char ch;
    do

```

```

    {
        scanf("%d%c",&a[len],&ch);
        len++;
    } while (ch!='\n');
    BubbleSort(a,n);
    for(int i=0; i<n; i++){
        sum+=a[i]*i;
    }
    printf("The minimum waiting time is %d.\n",sum);
    return 0;
}
//冒泡排序
void BubbleSort(int a[], int n){
    for(int i=0; i<n-1; i++){
        for(int j=0; j<n-i-1; j++){
            if(a[j]>a[j+1]){
                int temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
}

```

运行结果:

```

meat
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./最小等待时间
Enter the number of customers.
5
7 3 2 1 4
The minimum waiting time is 20.

```

字符统计

题目描述

有一段文章，共有 3 行文字，每行有 80 个字符。要求分别统计出其中英文大写字母、小写字母、空格、数字以及其他字符的个数。

代码:

```

#include<stdio.h>
#include<string.h>
int main(){
    char s[3][80];
    //大写字母个数，小写字母个数，空格个数，数字个数，其他字符个数
    int Upper=0, Lower=0, space=0, number=0, other=0;
    for(int i=0; i<3; i++){
        int j=0;
    }
}

```

```

do{
    scanf("%c",&s[i][j]);
    if(s[i][j]==' '){
        space++;
    }else if (s[i][j]>=65 && s[i][j]<=90)
    {
        Upper++;
    }else if (s[i][j]>=97 && s[i][j]<=122)
    {
        Lower++;
    }else if (s[i][j]>=48 && s[i][j]<=57)
    {
        number++;
    }else{
        other++;
    }
}while(s[i][j]!='\n');
}
printf("Upper: %d\n",Upper);
printf("Lower: %d\n",Lower);
printf("space: %d\n",space);
printf("number: %d\n",number);
printf("other: %d\n",other-3); //减去三个'\n'
return 0;
}

```

运行结果:

```

The maximum waiting time is 20.204
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./字符统计
Hello, I love you!
But,how about you?
yeah...
Upper: 3
Lower: 28
space: 5
number: 0
other: 7

```

杨辉三角

题目描述

杨辉三角，是二项式系数在三角形中的一种几何排列。在欧洲，这个表叫做帕斯卡三角形。帕斯卡（1623----1662）是在 1654 年发现这一规律的，比杨辉要迟 393 年，比贾宪迟 600 年。

代码:

```
#include<stdio.h>
```

```

int main(){
    int n;
    printf("Enter a number.\n");
    scanf("%d",&n);
    int a[n][n];
    //三角的两边赋值为 1
    for(int i=0; i<n; i++){
        a[i][i]=1;
        a[i][0]=1;
    }
    //从第三行开始，中间的每一个元素，等于该元素上面一个元素与左上元素之和
    for(int i=2; i<n; 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<n; i++){
        for(int j=0; j<=i; j++){
            printf("%6d",a[i][j]);
        }
        printf("\n");
    }
    return 0;
}

```

运行结果：

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./PascalTriangle
Enter a number.
10
    1
   1  1
  1  2  1
 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

```

成绩排序

题目描述

给出班里某门课程的成绩单，请你按成绩从高到低对成绩单排序输出，如果有相同分数则名字字典序小的在前。

输入

第一行为 n ($0 < n < 20$)，表示班里的学生数目；

接下来的 n 行，每行为每个学生的名字和他的成绩，中间用单个空格隔开。名字只包含字

母且长度不超过 20，成绩为一个不大于 100 的非负整数。

输出

n 行

把成绩单按分数从高到低的顺序进行排序并输出，每行包含名字和分数两项，之间有一个空格。

代码：

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
//定义学生结构体
struct student
{
    char name[20];
    float score;
    struct student *next_stu;
};
//定义头节点
struct student *stu_head;

void createStuList(int n);
void displayList();
void listSort(int n);

int main(){
    int n;
    printf("Input the number of students: ");
    scanf("%d",&n);
    createStuList(n);
    listSort(n);
    displayList();
    return 0;
}
//创建链表
void createStuList(int n){
    //current 指向当前节点，temp 用于指向上一个节点
    struct student *current, *temp;
    int i;
    //开辟一块内存空间给头节点
    stu_head=(struct student *)malloc(sizeof(struct student));
    if(stu_head==NULL){
        printf("Memory can not be allocated.");
    }else{
        printf("Input data for student 1: ");
        scanf("%s %f",stu_head->name,&stu_head->score);
```

```

    stu_head->next_stu=NULL;
    //将头节点地址赋给 temp
    temp=stu_head;
    for(i=2; i<=n; i++){
        //创建一个新的节点
        current=(struct student *)malloc(sizeof(struct student));
        if(current==NULL){
            printf("Memory can not be allocated.");
        }else{
            //给新节点赋值
            printf("Input data for student %d: ",i);
            scanf("%s %f",current->name,&current->score);
            current->next_stu=NULL;
            //将上一节点的 next 指针指向新节点
            temp->next_stu=current;
            //temp 指向新节点
            temp=current;
        }
    }
}

//链表排序
void listSort(int n){
    struct student *temp=stu_head;
    struct student *current, *next;
    if(temp->next_stu==NULL || temp==NULL){
        return;
    }else{
        //冒泡排序
        for(int i=0; i<n-1; i++){
            temp=stu_head;
            for(int j=0; j<n-i-1; j++){
                current=temp;
                next=temp->next_stu;
                //交换数据域
                if(current->score<next->score){
                    float score_t=current->score;
                    char name_t[20];
                    current->score=next->score;
                    next->score=score_t;
                    strcpy(name_t,current->name);
                    strcpy(current->name,next->name);
                    strcpy(next->name,name_t);
                }
            }
        }
    }
}

```

```

        temp=temp->next_stu;
    }
}
}
//打印链表
void displayList(){
    struct student *temp;
    if(stu_head==NULL){
        printf("List is empty.");
    }else{
        temp=stu_head;
        while (temp != NULL)
        {
            printf("name: %s score: %.1f\n",temp->name,temp->score);
            temp=temp->next_stu;
        }
    }
}
}

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./成绩排序
Input the number of students: 4
Input data for student 1: kitty 80
Input data for student 2: Hanmeimei 90
Input data for student 3: Joey 92
Input data for student 4: Tim 28
name: Joey score: 92.0
name: Hanmeimei score: 90.0
name: kitty score: 80.0
name: Tim score: 28.0

```

谁考了第 k 名

题目描述

在一次考试中，每个学生的成绩都不相同，现知道了每个学生的学号和成绩，求考第 k 名学生的学号和成绩。

输入

第一行有两个整数，分别是学生的人数 n ($1 \leq n \leq 100$)，和求第 k 名学生的 k ($1 \leq k \leq n$)。

其后有 n 行数据，每行包括一个学号（整数）和一个成绩（浮点数），中间用一个空格分隔。

输出

1 行

输出第 k 名学生的学号和成绩，中间用空格分隔。（注：请用 %g 输出成绩）

代码:

```

#include<stdio.h>
#include<stdlib.h>

```

```

#include<string.h>
#define MAXSIZE 100
typedef struct {
    char no[20];
    float score;
} stu;

typedef struct{
    stu *elem;
    int length;
} SqList;

int InitList(SqList *L);
void InputList(SqList *L, int n);
void DisplayList(SqList *L);
void SortList(SqList *L);

int main(){
    SqList *L;
    InitList(L);
    int n, k;
    printf("Input the total number of student and the rank you want: ");
    scanf("%d%d",&n,&k);
    InputList(L,n);
    SortList(L);
    printf("\n%s %g\n",L->elem[k-1].no,L->elem[k-1].score);
    //DisplayList(L);
    return 0;
}
//顺序表初始化
int InitList(SqList *L){
    L->elem=(stu *)malloc(sizeof(stu)*MAXSIZE);
    if (!L->elem)
    {
        exit(0);
    }
    L->length=0;
    return 1;
}
//输入顺序表的内容
void InputList(SqList *L, int n){
    for (int i = 0; i < n; i++){
        scanf("%s %f",L->elem[i].no,&L->elem[i].score);
        L->length++;
    }
}

```

```

    }
}

void DisplayList(SqList *L){
    for(int i=0; i<L->length; i++){
        printf("%s %g\n",L->elem[i].no,L->elem[i].score);
    }
}

//选择排序
void SortList(SqList *L){
    for(int i=0; i<L->length-1; i++){
        int max=i;
        for(int j=i+1; j<L->length; j++){
            if(L->elem[j].score>L->elem[max].score){
                max=j;
            }
        }
        float temp=L->elem[i].score;
        L->elem[i].score=L->elem[max].score;
        L->elem[max].score=temp;
        char no_t[20];
        strcpy(no_t,L->elem[i].no);
        strcpy(L->elem[i].no,L->elem[max].no);
        strcpy(L->elem[max].no,no_t);
    }
}

```

运行结果：

```

Input the total number of student and the rank you want: 5 3
2001 67.8
2002 90.3
2003 61
2004 68.4
2005 73.9

2004 68.4

```

整数奇偶排序

题目描述

给定 10 个整数的序列，要求对其重新排序。排序要求：

- 1.奇数在前，偶数在后；
- 2.奇数按从大到小排序；
- 3.偶数按从小到大排序。

输入

1 行：

输入一行，包含 10 个整数，彼此以一个空格分开，每个整数的范围是大于等于 0，小于等

于 100。

输出

1 行:

按照要求排序后输出一行，包含排序后的 10 个整数，数与数之间以一个空格分开。

代码:

```
#include<stdio.h>
```

```
#include<stdlib.h>
```

```
void DisplaySeq(int *);
```

```
void SortMax(int *, int);
```

```
void SortMin(int *, int);
```

```
int main(){
```

```
    int seq_origin[10], seq_sorted[10];
```

```
    int odd[10], even[10];
```

```
    for (int i = 0; i < 10; i++)
```

```
    {
```

```
        scanf("%d",&seq_origin[i]);
```

```
    }
```

```
    int count_o=0, count_e=0;
```

```
    //判断奇偶，分别放到奇数数组和偶数数组
```

```
    for(int i=0; i<10; i++){
```

```
        if(seq_origin[i]%2==0){
```

```
            even[count_e]=seq_origin[i];
```

```
            count_e++;
```

```
        }else if (seq_origin[i]%2!=0)
```

```
        {
```

```
            odd[count_o]=seq_origin[i];
```

```
            count_o++;
```

```
        }
```

```
    }
```

```
    //分别给奇数组和偶数组排序
```

```
    SortMax(odd, count_o);
```

```
    SortMin(even, count_e);
```

```
    //将奇数数组与偶数数组输入到新数组中
```

```
    for (int i = 0; i < count_o; i++)
```

```
    {
```

```
        seq_sorted[i]=odd[i];
```

```
    }
```

```
    for (int i =0; i < 10; i++)
```

```
    {
```

```
        seq_sorted[i+count_o]=even[i];
```

```
    }
```

```

    DisplaySeq(seq_sorted);
    return 0;
}

void DisplaySeq(int *seq){
    for (int i = 0; i < 10; i++){
        {
            printf("%d ",seq[i]);
        }
        printf("\n");
    }
}
//冒泡排序
void SortMax(int *seq, int n){
    for(int i=0; i<n-1; i++){
        for(int j=0; j<n-i-1; j++){
            if(seq[j]<seq[j+1]){
                int temp=seq[j];
                seq[j]=seq[j+1];
                seq[j+1]=temp;
            }
        }
    }
}
//选择排序
void SortMin(int *seq, int n){
    for(int i=0; i<n-1; i++){
        int min=i;
        for(int j=i+1; j<n; j++){
            if(seq[j]<seq[min]){
                int temp=seq[j];
                seq[j]=seq[min];
                seq[min]=temp;
            }
        }
    }
}

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验6$ ./整数奇偶排序
4 7 3 13 11 12 0 47 34 98
47 13 11 7 3 0 4 12 34 98

```

发现问题与分析：
备注：

说明：

- 1、 各项记录应完备详尽。
- 2、 各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、 可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称	函数递归				
实验目的：					
1. 掌握主调函数和被调函数之间的参数传递方式；					
2. 理解函数递归使用，使用递归解决实际问题。					
实验任务、步骤与结果：					
任务 1：求圆的周长与面积					
题目描述					
任务编制一个计算圆的面积与周长的程序，要求从键盘输入圆的半径，打印输出圆的周长与面积。					
输入					
1 行：					
圆的半径					
输出					
2 行：					
第 1 行，圆的周长（精确到小数点后 2 位）					
第 2 行，圆的面积（精确到小数点后 2 位）					
代码：					
#include<stdio.h>					
#define PI 3.1415926					
float circumference(float);					
float area(float);					
int main(){					
float radius;					
printf("Input the radius:\n");					
scanf("%f",&radius);					
printf("Circumference is %.2f\n",circumference(radius));					
printf("Area is %.2f\n",area(radius));					
return 0;					
}					
float circumference(float radius){					
return 2*PI*radius;					
}					
float area(float radius){					
return PI*radius*radius;					
}					
运行结果					

```
Input the radius:
2
Circumference is 12.57
Area is 12.57
```

任务 2：求 1~n 的阶乘和

题目描述

使用递归方法求 n 阶乘， $1 \leq n \leq 100$ ， $0! = 1$ ， $1! = 1$ 。

计算 $S=1\sim n$ 的阶乘和： $1! + 2! + 3! + \dots + n!$

输入

1 行：

n, $1 \leq n \leq 100$

输出

1 行：

S 的值

代码：

```
#include<stdio.h>
int factorial(int);
int main(){
    int n,sum=0;
    printf("Input a number:\n");
    scanf("%d",&n);
    for (int i = 1; i <= n; i++)
    {
        sum+=factorial(i);
    }
    printf("%d\n",sum);
    return 0;
}
int factorial(int n){
    if(n==0 || n==1){
        return 1;
    }else{
        return factorial(n-1)*n;
    }
}
```

运行结果：

```
(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud Files/248970588/
Input a number:
1
1
(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud Files/248970588/
Input a number:
2
3
(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud Files/248970588/
Input a number:
5
153
```

任务 3：汉诺塔 1

题目描述

汉诺塔（又称河内塔）问题是印度的一个古老的传说。开天辟地的神勃拉玛在一个庙里留下了三根金刚石的棒，第一根上面套着 64 个圆的金片，最大的一个在底下，其余一个比一个小，依次叠上去，庙里的众僧不倦地把它们一个个地从这根棒搬到另一根棒上，规定可利用中间的一根棒作为帮助，但每次只能搬一个，而且大的不能放在小的上面。面对庞大的数字(移动圆片的次数)18446744073709551615，看来，众僧们耗尽毕生精力也不可能完成金片的移动。

后来，这个传说就演变为汉诺塔游戏：

- 1.有三根杆子 A,B,C。A 杆上有若干碟子
- 2.每次移动一块碟子,小的只能叠在大的上面
- 3.把所有碟子从 A 杆全部移到 C 杆上

经过研究发现，汉诺塔的破解很简单，就是按照移动规则向一个方向移动金片：

如 3 阶汉诺塔的移动：A→C,A→B,C→B,A→C,B→A,B→C,A→C

此外，汉诺塔问题也是程序设计中的经典递归问题。

算法思路：

- 1.如果只有一个金片，则把该金片从源移动到目标棒，结束。
- 2.如果有 n 个金片，则把前 n-1 个金片移动到辅助的棒，然后把自己移动到目标棒，最后再把前 n-1 个移动到目标棒。

输入

1 行：

一个整数 N，表示 A 柱上有 N 个碟子。

输出

若干行：

移动的最少步骤

```
#include<stdio.h>
```

```
int Hanoi(int,char,char,char);
```

```
int main(){
```

```

    int n;
    char origin='A',middle='B',destination='C';
    printf("Input a number:\n");
    scanf("%d",&n);
    Hanoi(n,origin,middle,destination);
    return 0;
}
int Hanoi(int n, char origin, char middle,char destination){
    if(n==1){
        printf("%c To %c\n",origin,destination);
    }else{
        Hanoi(n-1,origin,destination,middle);
        printf("%c To %c\n",origin,destination);
        Hanoi(n-1,middle,origin,destination);
    }
    return 0;
}

```

运行结果:

```

Input a number:
3
A To C
A To B
C To B
A To C
B To A
B To C
A To C

```

任务 4: 汉诺塔 2

题目描述

模拟汉诺塔的移动方案，每次输出第 n 个盘子的移动方法，如 “ $a-2-c$ ” 表示从柱子 “ a ” 将编号为 “2” 的盘子移动到柱子 “ c ”。

输入

1 行:

一个整数 N ，表示 A 柱上有 N 个碟子。

输出

若干行:

每次的移动方案

最后一行，移动的总次数

```

#include<stdio.h>
int NumOfMoves=0;
int Hanoi(int,char,char,char);
int main(){
    int n;
    char origin='A',middle='B',destination='C';
    printf("Input a number:\n");

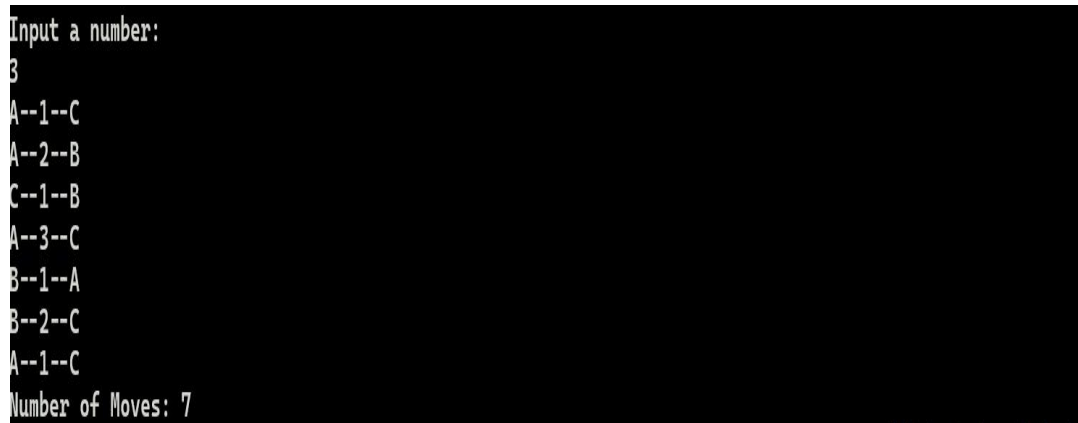
```

```

scanf("%d",&n);
Hanoi(n,origin,middle,destination);
printf("Number of Moves: %d\n",NumOfMoves);
return 0;
}
int Hanoi(int n, char origin, char middle,char destination){
    if(n==1){
        NumOfMoves++;
        printf("%c--%d--%c\n",origin,n,destination);
    }else{
        Hanoi(n-1,origin,destination,middle);
        NumOfMoves++;
        printf("%c--%d--%c\n",origin,n,destination);
        Hanoi(n-1,middle,origin,destination);
    }
    return 0;
}

```

运行结果:



```

Input a number:
3
A--1--C
A--2--B
C--1--B
A--3--C
B--1--A
B--2--C
A--1--C
Number of Moves: 7

```

任务 5：找出超过平均身高的人

题目描述

小明班上有 n 个同学，编程找出谁的身高超过全班的平均身高(整数)。

编写函数 `float inputdata(int a[],int n)`，输入 n 个的身高，数值存放在主函数的数组 a 中 (a 的元素个数最多为 100)，并返回平均身高；

编写函数 `int findhigh(int a[],int n,float ave)`，依次打印出每个超出平均身高 ave 的序号和身高，最后输出超出平均身高的人数。

输入

2 行:

第一行有一个整数 n ($1 < n < 50$)。第二行是 n 个整数，用空格隔开。

输出

3 行:

第一行为全家的平均身高（保留一位小数）；

第二行有若干个数，为超过平均身高的人的序号和身高厘米数(每项之前都有一个空格。)；

第三行为超过平均身高的同学人数。

```
#include<stdio.h>
float inputdata(int a[],int n);
int findhigh(int a[], int n, float ave);
int main(){
    int n, count;
    int a[50]={0};
    float average=0;
    printf("Input the number of classmates:\n");
    scanf("%d",&n);
    printf("Enter the height of classmates one by one:\n");
    average=inputdata(a,n);
    printf("AV=%.1f\n",average);
    count=findhigh(a,n,average);
    printf("\nTotal:%d\n",count);
    return 0;
}
float inputdata(int a[],int n){
    float sum=0;
    for (int i = 0; i < n; i++){
        scanf("%d",&a[i]);
        sum+=a[i];
    }
    return sum/n;
}
int findhigh(int a[], int n, float ave){
    int count=0;
    for(int i=0; i<n; i++){
        if(a[i]>ave){
            count++;
            printf("%d:%d\t",i+1,a[i]);
        }
    }
    return count;
}
```

运行结果:

```
Input the number of classmates:
5
Enter the height of classmates one by one:
177 188 193 175 190
AV=184.6
2:188 3:193 5:190
Total:3
```

任务 6: 分数化简

题目描述

编写函数 `reduction(int m,int n)`，调用最公约数函数 `gcd(int a,int b)`，实现分数化简，如果是真分数，写成 a/b 形式，如果是假分数，写成 $n+a/b$ 形式。

输入一个分数，输出该分数的最简分数。

输入

1 行:

两个整数，中间用 “/” 隔开。

输出

1 行:

约分后的最简分数，整数与分数部分使用 “+” 连接，分数中间 “/” 隔开。

```
#include<stdio.h>
int reduction(int,int);
int gcd(int,int);
int main(){
    int m,n;
    printf("Enter a fractional number:\n");
    scanf("%d/%d",&m,&n);
    reduction(m,n);
    return 0;
}
int reduction(int m, int n){
    if(m==n){
        printf("1\n");
    }else if (m>n)
    {
        int num;
        num=m/n;
        m=m-n;
        printf("%d+",num);
        return reduction(m,n);
    }else{
        int divisor=gcd(m,n);
        m=m/divisor;
        n=n/divisor;
```

```

        printf("%d/%d\n",m,n);
    }

}

int gcd(int m, int n){
    int temp;
    if(n>m){
        temp=m;
        m=n;
        n=temp;
    }
    if(m%n==0){
        return n;
    }else{
        return gcd(n,m%n);
    }
}

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/
Enter a fractional number:
20/16
1+1/4
(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/
Enter a fractional number:
16/12
1+1/3

```

发现问题与分析：在分数化简函数中，这里原来写成了 m 和 n 的值直接调用了 gcd() 来算，没有加一个中间变量。m=m/gcd(m,n);n=n/gcd(m,n); 导致出错。因为第二次调用 gcd() 时，m 值已经变了。后来重新定义了一个 divisor 变量，用于固定值。

备注：

说明：

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称	指针与数组（1）				

实验目的：

1. 掌握指针与变量地址的关系，掌握指针*与&运算基本方法；
2. 掌握指针和一维数组间的关系，掌握用指针变量引用一维数组元素的方法。

实验任务、步骤与结果：

任务 1：求圆的周长与面积

题目描述

任务编制一个计算圆的面积与周长的程序，要求从键盘输入圆的半径，打印输出圆的周长与面积。

代码：

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 int getaver(int *p);
4 int main(){
5     int a[10];
6     printf("Input 10 number:\n");
7     for(int i=0; i<10; i++){
8         scanf("%d",&a[i]);
9     }
10    int aver=getaver(a);
11    printf("Average is %d\n",aver);
12    return 0;
13 }
14 int getaver(int *p){
15     int sum=0;
16     for(int i=0; i<10; i++){
17         sum+=*p;
18         p++;
19     }
20     return sum/10;
21 }
```

运行结果

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验8$
Input 10 number:
1 2 3 4 5 6 7 8 9 10
Average is 5
```

任务 2：数字交换位置

题目描述

函数 void max_min_value (int *number,int n)实现将一个数组中最小数与第一个数对换，最大数与最后一个数对换。

输入

2 行:

第 1 行, n(n<20)

第 2 行, 输出数组中 n 个元素

输入

1 行:

交换后的数组, 每个元素之间用空格分开

```
1 #include<stdio.h>
2 void max_min_value(int *number, int n);
3 int main(){
4     int n;
5     printf("Enter a number:\n");
6     scanf("%d",&n);
7     int a[n];
8     printf("Enter %d number of array:\n",n);
9     for(int i=0; i<n; i++){
10         scanf("%d",&a[i]);
11     }
12     max_min_value(a,n);
13     printf("Result is: \n");
14     for(int i=0; i<n; i++){
15         printf("%d ",a[i]);
16     }
17     printf("\n");
18     return 0;
19 }
20 void max_min_value(int *number, int n){
21     int max, min, max_n=0, min_n=0;
22     int *temp, swap;
23     max=min=*number;
24     temp=number;
25     for(int i=0; i<n; i++){
26         if(*temp>max){
27             max=*temp;
28             max_n=i;
29         }else if(*temp<min){
30             min=*temp;
31             min_n=i;
32         }
33         temp++;
34     }
35     // 交换数值
36     *(number+min_n)=*number;
37     *number=min;
38     *(number+max_n)=*(number+n-1);
39     *(number+n-1)=max;
40 }
```

运行结果:

```
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/  
Enter a number:  
4  
Enter 4 number of array:  
3 2 5 4  
Result is:  
2 3 4 5
```

任务 3：数字向后移动

题目描述

有 n 个整数，编写函数 `int move_n(int* a,int n,int m)`，使前面各数顺序向后移 m 个位置，即最后 m 个数变成最前面 m 个数。

输入

3 行：

第 1 行， n

第 2 行， m

第 3 行， n 个整数（中间以空格隔开）

输入

1 行：

变序后的 n 个整数

```
#include<stdio.h>  
int move_n(int *a, int n, int m);  
int main(){  
    int n, m;  
    printf("Enter a number n:\n");  
    scanf("%d",&n);  
    printf("Enter a number of m:\n");  
    scanf("%d",&m);  
    int a[n];  
    for(int i=0; i<n; i++){  
        a[i]=i+1;  
    }  
    move_n(a,n,m);  
    for(int i=0; i<n; i++){  
        printf("%d ",a[i]);  
    }  
    printf("\n");  
    return 0;  
}  
int move_n(int *a, int n, int m){  
    int index=0;  
    for(int i=0; i<n; i++){  
        index=(i+m)%(n);  
        *(a+index)=i+1;  
    }  
    return 0;  
}
```

运行结果:

```
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter a number n:
6
Enter a number of m:
2
5 6 1 2 3 4
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter a number n:
8
Enter a number of m:
5
4 5 6 7 8 1 2 3
```

任务 4: 数字反序排列

题目描述

将数组 a 中 n 个整数按相反的顺序存放。编写函数 void invert(int *p,int n)实现以上功能。

输入

2 行:

第 1 行, 输入数组 a 的长度 n;

第 2 行, 输入数组 a。

输出

1 行:

反序后的数组 a。

```

1 #include<stdio.h>
2 void invert(int *p, int n);
3 int main(){
4     int n;
5     printf("Enter the length of array:\n");
6     scanf("%d",&n);
7     int a[n];
8     printf("Input the number of array:\n");
9     for(int i=0; i<n; i++){
10         scanf("%d",&a[i]);
11     }
12     invert(a,n);
13     for(int i=0; i<n; i++){
14         printf("%d ",a[i]);
15     }
16     printf("\n");
17     return 0;
18 }
19 void invert(int *p, int n){
20     int mid=n/2;
21     int swap;
22     for(int i=0; i<mid; i++){
23         swap=*(p+i);
24         *(p+i)=*(p+n-i-1);
25         *(p+n-i-1)=swap;
26     }
27 }

```

运行结果:

```

daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter the length of array:
4
Input the number of array:
6 8 3 7
7 3 8 6
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud
Enter the length of array:
5
Input the number of array:
7 6 3 4 8
8 4 3 6 7

```

任务 5: 约瑟夫问题

题目描述

n 个人围成一圈，顺序排号。从第 1 个人开始报数（从 1 到 m ），凡报到 m 的人退出圈子，问最后留下的是原来第几号。用函数和指针实现以上功能，返回数组中最后留下的元素序号。

输入

1 行:

n 和 m ，中间使用空格分开，分别表示总人数和该出局的报数。

输入

输出最后留下者的序号 x ，输出格式为: NO. x

```
10 #include<stdio.h>
11 int Josef(int *a, int n, int m);
12 int main(){
13     int n, m;
14     printf("Enter n and m:\n");
15     scanf("%d%d", &n, &m);
16     int a[n];
17     for(int i=0; i<n; i++){
18         a[i]=i+1;
19     }
20     int No;
21     No=Josef(a,n,m);
22     printf("NO.%d\n", No);
23     return 0;
24 }
25 int Josef(int *a, int n, int m){
26     int count=0, k=0, j=0, i=0;
27     //count用于统计0的个数，k用于表示当前序号，j用于累加每一个非0项，i正常累计用于控制循环进行
28     while(count<n-1){
29         k=i%n;
30         if(*(a+k)!=0){
31             //遇到非0则j+1
32             j++;
33             if(j%m==0){
34                 //j%m代表非零项每过m次判断一次
35                 *(a+k)=0;
36                 count++;
37             }
38         }
39         i++;
40     }
41     for(int h=0; h<n; h++){
42         if(*(a+h)!=0){
43             return h+1;
44         }
45     }
46     return 0;
47 }
```

运行结果:


```

daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud I
Enter n and m:
10 6
NO.3
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud I
Enter n and m:
20 2
NO.9
daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud I
Enter n and m:
8 5
NO.3

```

题目描述

有一数组 $a[N]$ ，需要对其局元素进行从高到低排序，请编写函数 `localsort(int *start, int len)`，实现数组指定位置开始后的 `len` 个元素从高到低排序。 $N < 100$ ， $start + len < \&a[N-1]$ 。

输入

2 行：

第 1 行, n, b, l ， n 表示数组元素个数， $n \leq N$, b 表示开始排序的下标， $1 < b \leq n$, l 表示从开始下标起排序元素长度, $1 \leq b + l \leq n$;

第 2 行, n 个数组元素。

输出

1 行，局部排序后的数组

【示例 1】

输入

```

10 2 5
1 2 3 4 5 6 7 8 9 10

```

输出

```

1 6 5 4 3 2 7 8 9 10

```

输入

```

5 1 5
6 8 7 4 2

```

输出

```

8 7 6 4 2

```

代码：

```

23 #include<stdio.h>
24 typedef int status;
25 #define OK 1
26 #define False 0
27 status localsort(int *start, int len);
28 int main(){

```



```

29     int a[100];
30     int n, b, l;
31     printf("输入数组元素个数:\n");
32     scanf("%d",&n);
33     printf("输入开始排序的下标: ");
34     scanf("%d",&b);
35     printf("输入排序的长度: ");
36     scanf("%d",&l);
37     printf("输入十个数组元素: ");
38     for(int i=0; i<n; i++){
39         scanf("%d",&a[i]);
40     }
41     localsort(&a[b-1],l);
42     for(int i=0; i<n; i++){
43         printf("%d ",a[i]);
44     }
45     printf("\n");
46     return 0;
47 }
48 status localsort(int *start, int len){
49     int *tmp;
50     tmp=start;
51     if(!tmp){
52         return False;
53     }
54     for(int i=0; i<len-1; i++){
55         int index;
56         int num=*(tmp+i);
57         for(int j=i+1; j<len; j++){
58             if(*(tmp+j)>num){
59                 num=*(tmp+j);
60                 index=j;
61             }
62         }
63         *(tmp+index)=*(tmp+i);
64         *(tmp+i)=num;
65     }
66     return OK;
67 }

```

运行结果:

```
(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud Files/248970588/计算机/程序设计基础/Experiment/实验8$ ./数组局部排列
输入数组元素个数：
10
输入开始排序的下标：2
输入排序的长度：5
输入十个数组元素：1 2 3 4 5 6 7 8 9 10
1 6 5 4 4 2 7 8 9 10
(base) daniel@DESKTOP-PG8AU51:/mnt/c/Users/Dadian/Documents/WPS Cloud Files/248970588/计算机/程序设计基础/Experiment/实验8$ ./数组局部排列
输入数组元素个数：
5
输入开始排序的下标：1
输入排序的长度：5
输入十个数组元素：8 7 6 4 2
Segmentation fault
```

结果错误

发现问题与分析：在数值交换程序中，一开始直接用*number 完成循环动作，然后想把 max 值给*number 时却无法正确输出。出现了 2000 多这样离谱的数值，初步怀疑是地址溢出。后来在程序中设了一个*temp，把 number 地址赋给 temp，赋值比较过程由 temp 进行。最后的结果再赋给*number。

备注：

说明：

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称	指针与数组（2）				
实验目的：					
1. 理解字符串存储基本原理以及字符串与指针关系；					
2. 掌握指针操作字符串的基本方法。					
3. 掌握指针与不同数据类型、数组、字符串的关系；					
4. 使用指针解决综合问题。					
实验任务、步骤与结果：					
任务 1：间隔输出字母					
题目描述					
编写程序，主函数从键盘上接收一个字符串，调用 printchar（）函数，将作字符串中字母间隔打印出来，如输入 Computer，则输出 Cmue。输入 C1o2m3p4u5t6e7r8，则输出 Cmue。					
函数原型：int printchar(char *s)					
输入					
1 行：					
输入一行不大于 20 个字符的字符串					
输出					
1 行：					
输出间隔打印的字母					
【示例 1】					
输入					
student					
输出					
suet					
【示例 2】					
输入					
C1o2m3p4u5t6e7r8					
输出					
Cmue					
【示例 3】					
输入					
1234567890					
输出					
代码：					

```

1 #include<stdio.h>
2 int printchar(char *s);
3 int main(){
4     char str[20];
5     printf("Enter a word.\n");
6     fgets(str, 20, stdin);
7     printchar(str);
8     printf("\n");
9     return 0;
10 }
11
12 int printchar(char *s){
13     char *str;
14     str=s;
15     while(*str!='\0'){
16         printf("%c",*str);
17         str=str+2;
18     }
19     return 0;
20 }

```

测试结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./间隔输出字母
Enter a word.
student
suet
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./间隔输出字母
Enter a word.
computer
cmue
♦♦
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./间隔输出字母
Enter a word.
abcdefghi
acegi
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$

```

任务 2: 单词统计

题目描述

主函数中输入字符串，编写函数 `int wordCount(char *s)`，实现字符串单词统计功能。

输入

1 行:

一个包括多个单词的字符串，单词之间由空格格式。

输出

1 行:

统计出的单词个数

【示例 1】

输入

I am a student

输出

4

【示例 2】

输入

My school is Jiangsu University Of Technology

输出

7

【示例 3】

输入

I like eating fruits

输出

4

代码:

```
1 #include<stdio.h>
2 int wordCount(char *s);
3 int main(){
4     char str[100];
5     printf("Enter a sentence:\n");
6     fgets(str,100,stdin);
7     printf("单词个数: %d.\n",wordCount(str));
8     return 0;
9 }
10 int wordCount(char *s){
11     int count=0;
12     char *str;
13     str=s;
14     if(str!=NULL){
15         count++;
16     }
17     while(*str!='\0'){
18         if(*str==' '){
19             count++;
20         }
21         str++;
22     }
23     return count;
24 }
```

运行结果:

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./单词统计
Enter a sentence:
I like eating fruits
单词个数: 4.
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./单词统计
Enter a sentence:
My school is Jiangsu University Of Technology
单词个数: 7.
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./单词统计
Enter a sentence:
I am a student
单词个数: 4.
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$
```

任务 3: 字符串翻转

输入一行不超过 80 个字符的字符串, 将其翻转后输出。

函数原型: `int strReverse(char *s)`

输入

1 行:

不超过 80 个字符的字符串

输出

1 行:

翻转后的字符串。

【示例 1】

输入

I love China

输出

anihC evol I

【示例 2】

输入

excited

输出

Deticxe

代码:

```
1 #include<stdio.h>

1 #include<stdio.h>
2 #include<string.h>
3 int strReverse(char *s);
4 int main(){
5     char str[80];
6     printf("Enter a sentence:\n");
7     fgets(str,80,stdin);
8     int count;
```

```

9          count=strReverse(str);
10         for(int i=0; i<count;
i++){
11             printf("%c",str[i]);
12         }
13         printf("\n");
14         return 0;
15     }
16 int strReverse(char *s){
17     char *str;
18     int count=0;
19     count=strlen(s);
20     str=s+count-1;
21     while(s<str){
22         char tmp;
23         tmp=*str;
24         *(str--)=*s;
25         *(s++)=tmp;
26     }
27     return count;
28 }

```

运行结果:

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./字符串翻转
Enter a sentence:
I love China

anihC evol I
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$

```

任务 4: 单词逆序

题目描述

输入一行不超过 10 个单词的英文字符串，将其中单词逆顺序输出。

函数原型: `int wordReverse(char *s, char wd[10][20])`，接收字符串，并把串中单词提取出来存放在二维数组 `wd[10][20]` 中，函数返回单词数量。逆序后的单词，在主函数中输出。

输入

1 行:

一个不超过 10 个单词的字符串，单词之间由空格分开

输出

1 行:

逆序排列后的单词，单词之间由空格分开，最后一个单词后换行。

【示例1】

输入

I love China

输出

China love I

【示例2】

输入

Hello

输出

Hello

【示例2】

输入

I'am a Chinese student.

输出

student. Chinese a I'am

代码:

```
1 #include<stdio.h>
2 #include<string.h>
3 int wordReverse(char *s, char wd[10][20]);
4 int main(){
5     char str[100];
6     printf("Enter a sentence:\n");
! 7     gets(str);
8     char wd[10][20];
9     int row=wordReverse(str,wd);
10    for(int i=row; i>=0; i--){
11        printf("%s",wd[i]);
12    }
13
14    return 0;
15 }
16
17 int wordReverse(char *s, char wd[10][20]){
18     int count;
19     count=strlen(s);
20     int row=0;
21     int j=0;
22     while(*s!='\0'){
23         (*(wd+row)+j)=*s;
24         j++;
25         if(*s==' '){
26             row++;
27             j=0;
```



```

28         }
29         s++;
30     }
31     return row;
32 }

```

运行结果：

```

(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./单词逆序
Enter a sentence:
I love China
China love I
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./单词逆序
Enter a sentence:
Hello
Hello
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$ ./单词逆序
Enter a sentence:
I'm a Chinese student
studentChinese @Ia I'm
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验9$

```

发现问题与分析：在字符串翻转程序中，原代码部分为 `for(int i=0; i<count; i++){*s=*str; s++; str--;}` 结果导致只能翻转第一个单词，其他单词也没有出现。经过参考实例代码，发现应该在交换到字符串一半位置的时候就应该停止，故更改判断条件，并引入中间变量。

单词逆序一题中，输出结果总是在第一个单词后面换行，推测是由于 `fgets` 捕捉了换行符。换成 `gets` 后虽然没有了第一个单词换行的问题，但每个单词前面多了一个 `@` 符号，暂时没有解决。

备注：

说明：

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。

学号	2021144301	姓名	王大殿	实验日期	
实验名称	结构体与单链表				
实验目的：					
1. 掌握结构体类型的说明、结构体变量(数组)的定义及初始化方法；					
2. 掌握结构体变量成员的引用；。					
实验任务、步骤与结果：					
任务 1：求圆的周长与面积					
题目描述					
1. 学生成绩录入与输出					
设计结构体并声明变量，存储学生学号，姓名，语，数，外三门课程成绩。					
函数					
DataInput()					
DataOutput()					
分别完成数据输与输出					
【输入要求】					
N+1 行：					
第 1 行，输入一个正整数 N（N <= 100），表示学生人数。					
第 2 至接着输入 N+1 行，第 1~N 个学生信息，每行格式如下：					
学号 姓名 分数 1 分数 2 分数 3					
分数是一个非负整数，且小于等于 100；					
姓名为一个连续的字符串，中间没有空格，长度不超过 20。					
【输出要求】					
N+1 行：					
第 1 行，表头，第 2~N+1 行，学生成绩信息。					
成绩精确到小数点后面一位					
学号 姓名 分数之间的间距为一个 Tab					
代码：					
#include<stdio.h>					
#include<malloc.h>					
#include<string.h>					

```
typedef int Status;

//学生信息内容
typedef struct{
    char stu_ID[10];
    char name[20];
    float score[3];
}student;

typedef student ElemType;

//链表结构
typedef struct LNode{
    ElemType data;
    struct LNode *next;
}LNode, *LinkList;

//链表初始化
Status InitList(LinkList *L);

//显示学生信息
void Output(ElemType *e);

//获取学生信息
Status GetElem(LinkList L, int i, ElemType *e);

//在指定位置插入某个学生信息
Status ListInsert(LinkList L, int i, ElemType e);

//根据名字进行查找
Status Search(LNode L, char str[], LinkList *p);

//删除指定位置的学生信息
Status ListDelete(LinkList L, int i);

//输入学生信息
void Input(ElemType *e);

int main(){
    LinkList L;
    int n, choose; //学生个数以及菜单序号
    ElemType a, b; //用于暂存学生信息
    printf("\n1.构造链表\n");
    printf("2.输入学生信息\n");
```

```

printf("3.显示学生表信息\n");
printf("9.退出\n\n");
while(1){
    printf("请选择: ");
    scanf("%d",&choose);
    if(choose==9) break;
    switch(choose){
        case 1: if(InitList(&L)){
                    printf("成功建立链表\n");
                }else{
                    printf("失败");
                }
                break;
        case 2: printf("请输入要录入的学生人数: ");
                scanf("%d",&n);
                for(int i=1; i<=n; i++){
                    printf("第%d 个学生: \n",i);
                    Input(&a);
                    ListInsert(L, i, a);
                }
                break;
        case 3: for(int i=1; i<=n; i++){
                    GetElem(L,i,&b);
                    Output(&b);
                }
                break;
    }
}

Status InitList(LinkList *L){
    (*L)=(LinkList)malloc(sizeof(LNode));
    (*L)->next=NULL;
    return 1;
}

void Output(ElemType *e){
    printf("学号: %s\t 姓名: %s\t 成绩 1: %.1f\t 成绩 2: %.1f\t 成绩 3: %.1f\n",
        e->stu_ID,e->name,e->score[0],e->score[1],e->score[2]);
}

Status GetElem(LinkList L, int i, ElemType *e){
    LinkList P;
    P=L->next;
    int j=1;
    while (P&& j<i){

```

```

        //P 不为空, j 小于 i
        P=P->next;
        j++;
    }
    if(!P || j>i){
        //如果 P 为空, 或 j 大于 i
        return 0;
    }
    *e=P->data;
    return 1;
}
Status ListInsert(LinkList L, int i, ElemType e){
    LinkList P, S;
    P=L;
    int j=0;
    while(P && j<i-1){
        //j 小于 i-1, 空出 i 的位置
        P=P->next;
        j++;
    }
    if(!P || j>i-1){
        return 0;
    }
    S=(LinkList)malloc(sizeof(LNode));
    S->data=e; //把 e 的值赋给新节点的数据部分
    S->next=P->next; //把第 i-1 个节点的下一个节点位置赋给新建节点的 next
    P->next=S; //第 i-1 的 next 指向新建节点
    return 1;
}
void Input(ElemType *e){
    printf("学号: "); scanf("%s",e->stu_ID);
    printf("姓名: "); scanf("%s",e->name);
    printf("成绩 1: "); scanf("%f",&e->score[0]); //这里一定要用取地址符, 原因未知?
    printf("成绩 2: "); scanf("%f",&e->score[1]);
    printf("成绩 3: "); scanf("%f",&e->score[2]);
    printf("输入完成\n");
}

```

运行结果

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验10$ ./学生信息管理
```

```
1.构造链表
2.输入学生信息
3.显示学生表信息
9.退出
```

```
请选择: 1
```

```
成功建立链表
```

```
请选择: 2
```

```
请输入要录入的学生人数: 2
```

```
第1个学生:
```

```
学号: 2001
```

```
姓名: wdd
```

```
成绩1: 10
```

```
成绩2: 29
```

```
成绩3: 12
```

```
输入完成
```

```
第2个学生:
```

```
学号: 2002
```

```
姓名: wer
```

```
成绩1: 29
```

```
成绩2: 34
```

```
成绩3: 12
```

```
输入完成
```

```
请选择: 3
```

```
学号: 2001      姓名: wdd      成绩1: 10.0      成绩2: 29.0      成绩3: 12.0
```

```
学号: 2002      姓名: wer      成绩1: 29.0      成绩2: 34.0      成绩3: 12.0
```

```
请选择: 9
```

3 约瑟夫（链表）

用链表实现约瑟夫问题：13 个人围成一圈，从第 1 个人开始顺序报号 1，2，3。凡报到 3 者退出圈子。找出最后留在圈子中的人原来的序号。

代码：

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
typedef struct node {
```

```
    int number;
```

```
    struct node * next;
```

```
}person;
```

```
person * initLink(int n) {
```

```
    int i = 0;
```

```
    person * head = NULL, *cyclic = NULL;
```

```
    head = (person*)malloc(sizeof(person));
```

```
    head->number = 1;
```

```
    head->next = NULL;
```

```
    cyclic = head;
```

```
    for (i = 2; i <= n; i++) {
```

```
        person * body = (person*)malloc(sizeof(person));
```

```
        body->number = i;
```

```
        body->next = NULL;
```

```
        cyclic->next = body;
```

```
        cyclic = cyclic->next;
```

```
    }
```

```
    cyclic->next = head;//首尾相连
```

```

        return head;
    }

    void findAndKillK(person * head, int k, int m) {
        person * p = NULL;
        person * tail = head;
        //找到链表第一个结点的上一个结点，为删除操作做准备
        while (tail->next != head) {
            tail = tail->next;
        }
        p = head;
        //找到编号为 k 的人
        while (p->number != k) {
            tail = p;
            p = p->next;
        }
        //从编号为 k 的人开始，只有符合 p->next==p 时，说明链表中除了 p 结点，所有
        //编号都出列了，
        while (p->next != p) {
            int i = 0;
            //找到从 p 报数 1 开始，报 m 的人，并且还要知道数 m-1de 人的位置 tail，方
            //便做删除操作。
            for (i = 1; i < m; i++) {
                tail = p;
                p = p->next;
            }
            tail->next = p->next; //从链表上将 p 结点摘下来
            printf("出列人的编号为:%d\n", p->number);
            free(p);
            p = tail->next; //继续使用 p 指针指向出列编号的下一个编号，游戏继续
        }
        printf("出列人的编号为:%d\n", p->number);
        free(p);
    }

    int main() {
        int n = 0, k = 0, m = 0;
        person * head = NULL;
        printf("输入圆桌上的人数:");
        scanf("%d", &n);
        head = initLink(n);
        printf("从第几个人开始报数(k>1 且 k<=%d): ", n);
        scanf("%d", &k);
        printf("数到几的人出列: ");
    }

```

```
scanf("%d", &m);  
findAndKillK(head, k, m);  
return 0;  
}  
运行结果:
```

```
(base) daniel@DESKTOP-PG8AU51:~/WPS/计算机/程序设计基础/Experiment/实验10$ ./约瑟夫_链表  
输入圆桌上的人数:10  
从第几个人开始报数(k>1且k<10): 3  
数到几的人出列: 5  
出列人的编号为:7  
出列人的编号为:2  
出列人的编号为:8  
出列人的编号为:4  
出列人的编号为:1  
出列人的编号为:10  
出列人的编号为:3  
出列人的编号为:6  
出列人的编号为:9  
出列人的编号为:5
```

发现问题与分析:

备注:

说明:

- 1、各项记录应完备详尽。
- 2、各栏目如果长度不够可以自行调整大小，但应注意排版的美观。
- 3、可将实现过程的代码、运行效果等内容以贴图方式放在相应栏目。