

夢想 從這裡 開始

C 语言程序设计 题库及参考答案

Part I 程序设计



Version: 2013.9

Foreword

评分标准

每题程序的得分为算法正确率与编程规范系数的乘积。

程序算法正确率： 编译通过并且结果基本正确，边界条件未处理好则酌情扣分。
即评分时只看结果是否正确，原则上不检查程序算法内容。
建议尽可能使用题中给出的可用素材。

编程规范系数： 系数取值默认为 1，不符合编程规范的程序系数则酌情减少。
若出现 3 个及以上的地方不符合编程规范，则系数取值为 0。

复制文件 — Microsoft Word

有的题目提供了相关文件，可复制（有的文件还可直接打开）。

右侧提供包含所有文件的压缩包“Attachment Pack.zip”。

复制方法： 选择文件图标，按一般复制操作进行。



Attachment Pack.zip

获取文件 — Adobe PDF


相关文件以附件形式提供，文件名为“附件.docx”。

获取方法： 保存或打开附件内容，在 Word 环境下按上述方法操作。

代码检查

代码检查工具在需要时辅助检查程序的编制是否符合要求。

某些题已提供部分代码并禁止修改已有内容，并对新增代码行数及行长有限制。

 该工具可复制或双击直接运行（chkUserCode.exe）。



chkUserCode.exe

? 帮助资源

建议访问帮助系统（快捷键：F1）以查看帮助内容及示例程序。

如帮助系统失效，可访问此链接：<http://msdn.microsoft.com/zh-CN/vstudio>。

+ 附加内容

五个附录：

C 运算符优先级与结合性表、ASCII 值对照表、C 数据类型及其取值范围、C 关键字及正式考试试题。

参考答案包

所有程序设计题答案（含附录）的源文件以及必要的测试文件。

右侧源程序包可复制。



Extra.zip

答案仅供参考！《题库及参考答案》由两部分组成

Part I 程序设计（本册）

Part II 程序修改与调试

Table of Contents

DIFFICUTLY ★

P122	1	P112	33	P330	74
P123	1	P115	34	P716	75
P709	2	P116	36	P745	76
P710	3	P117	37	P747	77
P711	3	P118	37	P803	78
P712	4	P119	38	P806	79
P720	5	P120	39	P816	81
P721	5	P121	40	P831	82
P722	6	P221	41	P239	83
P727	7	P224	42	P241	84
P728	7	P225	43	P243	85
P729	8	P226	44	P244	87
P730	9	P227	45	P245	88
P733	9	P228	46	P246	89
P734	10	P229	47	P247	91
P708	10	P230	47	P248	92
P735	11	P231	48	P249	93
P736	12	P232	49	P250	94
P111	13	P233	50	P251	96
P113	13	P320	51	P252	97
P114	14	P323	52	P253	98
P124	16	P714	54	P254	100
P125	17	P718	55	P255	102
P126	18	P732	55	P256	103
P127	18	P743	56	P792	105
P238	19	P744	57	P811	106
P702	19	P750	58	P812	107
P703	21	P752	59	P813	109
P713	22	P753	60	P814	110
P723	23	P800	61	P817	111
P731	24	P830	62	P820	112
P737	25	P215	63	P821	114
P738	25	P223	64	P822	115
P749	26	P257	65	P824	116
P754	27	P321	66	P240	118
P755	29	P324	67	P242	119
P778	30	P325	68	P789	120
P827	32	P327	70	P793	122
P828	33	P328	72	P796	123

DIFFICUTLY ★★

P219	125
P220	126
P222	127
P717	127
P725	128
P211	129
P212	130
P213	132
P214	134
P216	135
P217	136
P218	137
P234	138
P235	139
P236	140
P237	141
P724	142
P739	143
P740	144
P742	145
P748	146
P770	147
P818	148
P823	150
P311	151
P701	153
P704	154
P705	155
P715	157
P768	158
P771	161
P773	162
P776	163
P808	165
P825	166
P421	169
P422	172
P423	175

P707	178
P726	179
P741	180
P746	181
P751	182
P775	184
P777	186
P804	187
P761	189
P762	191
P763	192
P780	194
P782	195
P783	197
P784	199
P785	200
P786	202
P787	205
P788	206
P790	208
P791	210
P795	212
P797	213
P798	215
P801	216
P807	218
P826	220
P317	222
P312	224
P314	226
P318	227
P319	229
P322	230
P764	232
P802	234
P805	236
P809	238

DIFFICUTLY ★★★

P313	240
P329	241
P412	243
P415	245
P419	247
P719	248
P765	249
P767	251
P769	252
P772	254
P815	255
P315	256
P502	258
P766	260
P781	262
P794	264

DIFFICUTLY ★★★★

P316	267
P507	269
P511	274
P512	276
P513	277
P514	278
P515	280
P706	284
P774	285

Appendices

Appendix A C Operators	289
Appendix B ASCII Charmap	290
Appendix C Data Types	291
Appendix D C Keywords	291
Appendix E Exam	292

P122

DIFFICUTLY ★

从键盘读入 4 个数 num_1 、 num_2 、 num_3 、 num_4 ，输出： $(num_1 \div num_2 \text{ 的余数}) \times num_3 + num_4$ ，不需考虑 num_2 为 0 和计算结果溢出的情况。要求输出的结果中，整数部分宽度为 8（不足 8 时以 0 补足），小数部分宽度为 7。

可用素材： `printf("请输入4个数: ")`
`printf("\n计算结果为: ...`

程序的运行效果应类似地如图 122.1 和图 122.2 所示，金色部分是从键盘输入的内容。



图 122.1



图 122.2

参考答案：

```
#include <stdio.h>

int main(void)
{
    int num1, num2, num4;
    double num3, result;

    printf("请输入4个数: ");
    scanf("%d%d%lf%d", &num1, &num2, &num3, &num4);
    result = num3 * (num1 % num2) + num4;
    printf("\n计算结果为: %016.7f\n", result);

    return 0;
}
```

P123

DIFFICUTLY ★

从键盘读入 4 个数 num_1 、 num_2 、 num_3 、 num_4 ，输出： $num_1 + (num_2 \div num_3 \text{ 的余数}) \times num_4$ ，不需考虑 num_3 为 0 和计算结果溢出的情况。要求输出的结果中，整数部分宽度为 7（不足 7 时以 0 补足），小数部分宽度为 5。

可用素材： `printf("请输入4个数: ")`
`printf("\n计算结果为: ...`

程序的运行效果应类似地如图 123.1 和图 123.2 所示，金色部分是从键盘输入的内容。



图 123.1



图 123.2

参考答案:

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

```
int main(void)
```

```
{
```

```
    int num1, num2, num3;
```

```
    double num4, result;
```

```
    printf("请输入4个数: ");
```

```
    scanf("%d%d%d%lf", &num1, &num2, &num3, &num4);
```

```
    result = num1 + (num2 % num3) * num4;
```

```
    printf("\n计算结果为: %013.5f\n", result);
```

```
    return 0;
```

```
}
```

P709

DIFFICUTLY ★

输入一个华氏温度，要求输出摄氏温度，计算公式为 $C = \frac{5(F - 32)}{9}$ 。

可用素材: `printf("Input the degree: ")`

`printf("\nF(...)=C(...)...")`

程序的运行效果应类似地如图 709.1 所示，金色部分是从键盘输入的内容。



图 709.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    float F0, C0;

    printf("Input the degree: ");
    scanf("%f", &F0);
    C0 = 5 * (F0 - 32) / 9;
    printf("\nF(%.2f)=C(%.2f)\n", F0, C0);

    return 0;
}
```

P710

DIFFICUTLY ★

输入一个小写英文字母，首先输出它及其 ASCII 码，然后输出其对应的大写字母及其 ASCII 码。

可用素材: `printf("Input a lowercase letter: ")`
`printf("\n...()...")`

程序的运行效果应类似地如图 710.1 所示，金色部分是从键盘输入的内容。



图 710.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    char small, cap;

    printf("Input a lowercase letter: ");
    small = getchar();
    cap = small - 32;
    printf("\n%c(%d)\n%c(%d)\n", small, small, cap, cap);

    return 0;
}
```

P711

DIFFICUTLY ★

用 `scanf()` 输入圆半径 r 、圆柱高 h ，求圆周长 C ($C = 2\pi r$)、圆面积 S ($S = \pi r^2$)、圆柱体积 V ($V = \pi r^2 h$)。

注：本题中规定圆周率 (π) 取值为 3.14。

可用素材: `printf("Input: ")`

```
printf("\nC1=...\nS=...\nV=...
```

程序的运行效果应类似地如图 711.1 所示，金色部分是从键盘输入的内容。



图 711.1

参考答案:

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

```
#define PI 3.14
```

```
int main(void)
```

```
{
```

```
    float r0, h0, C0, S0;
```

```
    printf("Input: ");
```

```
    scanf("r=%f,h=%f", &r0, &h0);
```

```
    C0 = (float)(2 * r0 * PI);
```

```
    S0 = (float)(r0 * r0 * PI);
```

```
    printf("\nC1=%.2f\nS=%.2f\nV=%.2f\n", C0, S0, S0 * h0);
```

```
    return 0;
```

```
}
```

P712

DIFFICUTLY ★

判断输入的正整数是否既是 5 又是 7 的整倍数，若是，则输出 “Yes.”；否则输出 “No.”。

可用素材: `printf("请输入4个数: ")`

`printf("\n计算结果为: ...`

程序的运行效果应类似地如图 712.1 和图 712.2 所示，金色部分是从键盘输入的内容。



图 712.1



图 712.2

参考答案:

```
#include <stdio.h>

int main(void)
{
    int num;

    printf("Please input an integer: ");
    scanf("%d", &num);
    if (num % 5 == 0 && num % 7 == 0)
    {
        printf("\nYes.\n");
    }
    else
    {
        printf("\nNo.\n");
    }

    return 0;
}
```

P720

DIFFICUTLY ★

输入实型数据 a 、 b ，然后输出 a 、 b 的值。

可用素材: `printf("please input two numbers: ")`
`printf("\na=...,b=...")`

程序的运行效果应类似地如图 720.1 所示，金色部分是从键盘输入的内容。



图 720.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    float a0, b0;

    printf("please input two numbers: ");
    scanf("%f,%f", &a0, &b0);
    printf("\na=%.6f,b=%.6f\n", a0, b0);

    return 0;
}
```

P721

DIFFICUTLY ★

从键盘输入 x 、 y 、 z 的值，编写程序输出以下表达式的值: $x + z \% 3 * (\text{int})(x + y) \% 2 / 4$ 。

可用素材: `printf("please input x,y,z: ")`
`printf("\nResult = ...`

程序的运行效果应类似地如图 721.1 所示，金色部分是从键盘输入的内容。



图 721.1

参考答案:

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

```
int main(void)
{
    float num1, num2, result;
    int num3;

    printf("please input x,y,z: ");
    scanf("%f,%f,%d", &num1, &num2, &num3);
    result = num1 + num3 % 3 * (int)(num1 + num2) % 2 / 4;
    printf("\nResult = %.6f\n", result);

    return 0;
}
```

P722

DIFFICUTLY ★

从键盘输入一日期，年月日之间以“-”分隔，并以同样的形式但以“/”作分隔符输出。

可用素材: `printf("\nplease input a date: ")`
`printf("\nthe date is: ...`

程序的运行效果应类似地如图 722.1 所示，金色部分是从键盘输入的内容。



图 722.1

参考答案:

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

```
int main(void)
{
    int year, month, day;

    printf("\nplease input a date: ");
    scanf("%d-%d-%d", &year, &month, &day);
    printf("\nthe date is: %d/%02d/%02d\n", year, month, day);
}
```

```
    return 0;
}
```

P727

DIFFICUTLY ★

输入两个整数，输出这两个整数的和。

可用素材： `printf("please input data: ")`
`printf("\nResult: ... +... =...")`

程序的运行效果应类似地如图 727.1 所示，金色部分是从键盘输入的内容。



图 727.1

参考答案：

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

```
int main(void)
```

```
{
```

```
    int num1, num2;
```

```
    printf("please input data: ");
```

```
    scanf("%d%d", &num1, &num2);
```

```
    printf("\nResult:  %2d + %2d = %2d\n", num1, num2, num1 + num2);
```

```
    return 0;
```

```
}
```

P728

DIFFICUTLY ★

已知某产品单价是 30，输入其数量是 *num*，输出其总价。

可用素材： `printf("please input num: ")`
`printf("\ntotal=...")`

程序的运行效果应类似地如图 728.1 所示，金色部分是从键盘输入的内容。



图 728.1

参考答案：

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

```

#define PRICE 30

int main(void)
{
    int num;

    printf("please input num: ");
    scanf("%d", &num);
    printf("\ntotal=%d\n", num * PRICE);

    return 0;
}

```

P729

DIFFICUTLY ★

输入 x 、 y 两个整数，输出其中较大的数。

可用素材： printf("please input x,y: ")
 printf("\n... is bigger\n"...

程序的运行效果应类似地如图 729.1 所示，金色部分是从键盘输入的内容。



图 729.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int x0, y0, max;

    printf("please input x,y: ");
    scanf("%d,%d", &x0, &y0);
    if (x0 > y0)
    {
        max = x0;
    }
    else
    {
        max = y0;
    }
    printf("\n%d is bigger\n", max);

    return 0;
}

```

P730

DIFFICUTLY ★

输入两个整数，如果相等输出“yes”，否则输出“no”。

可用素材： `printf("please input x y: ")`
`printf("\nyes")`、`printf("\nno")`

程序的运行效果应类似地如图 730.1 所示，金色部分是从键盘输入的内容。



图 730.1

参考答案：

`#include <stdio.h>`

```
int main(void)
{
    int x0, y0;

    printf("please input x y: ");
    scanf("%d%d", &x0, &y0);
    if (x0 == y0)
    {
        printf("\nyes\n");
    }
    else
    {
        printf("\nno\n");
    }

    return 0;
}
```

P733

DIFFICUTLY ★

从键盘读入一个任意字符，输出该字符 ASCII 的十六进制值。

可用素材： `printf("Input a character: ")`
`printf("\nAscii('...') = 0x...\n")`

程序的运行效果应类似地如图 733.1 所示，金色部分是从键盘输入的内容。



图 733.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    char ch;

    printf("Input a character: ");
    ch = getchar();
    printf("\nAscii('%c') = 0x%x\n", ch, ch);

    return 0;
}
```

P734

DIFFICUTLY ★

从键盘输入一个正方体的边长（整型），计算该正方体的体积和表面积。

可用素材: `printf("Input a side of cube: ")`
`printf("\nThe volume of cube is ..., the surface area of cube is %d.\n"...`

程序的运行效果应类似地如图 734.1 所示，金色部分是从键盘输入的内容。



图 734.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int a0, V0, S0;

    printf("Input a side of cube: ");
    scanf("%d", &a0);
    V0 = a0 * a0 * a0;
    S0 = 6 * a0 * a0;
    printf("\nThe volume of cube is %d, the surface area of cube is %d.\n", V0, S0);

    return 0;
}
```

P708

DIFFICUTLY ★

输入一个三位正整数，然后逆序输出。如输入 123，则输出 321。

可用素材: `printf("Input an integer: ")`
`printf("\nThe result is ...`

程序的运行效果应类似地如图 708.1 所示，金色部分是从键盘输入的内容。



图 708.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int num, bai, shi, ge;

    printf("Input an integer: ");
    scanf("%d", &num);
    if (num >= 100 && num <= 999)
    {
        ge = num % 10;
        shi = num / 10 % 10;
        bai = num / 100;
        printf("\nThe result is %d%d%d\n", ge, shi, bai);
    }
    else
    {
        printf("Invalid value\n");
    }

    return 0;
}
```

P735

DIFFICUTLY ★

从键盘输入一个正三角形的边长（整型），计算该三角形的面积和周长。

注：根据“海伦—秦九韶”公式， $area = \sqrt{p(p-a)(p-b)(p-c)}$ ，其中 $p = \frac{a+b+c}{2}$ ， a 、 b 、 c 为三角形的 3 条边长。

可用素材： `printf("Input a side of triangle: ")`
`printf("\nThe area of triangle is ..., the circle of triangle is`

程序的运行效果应类似地如图 735.1 所示，金色部分是从键盘输入的内容。

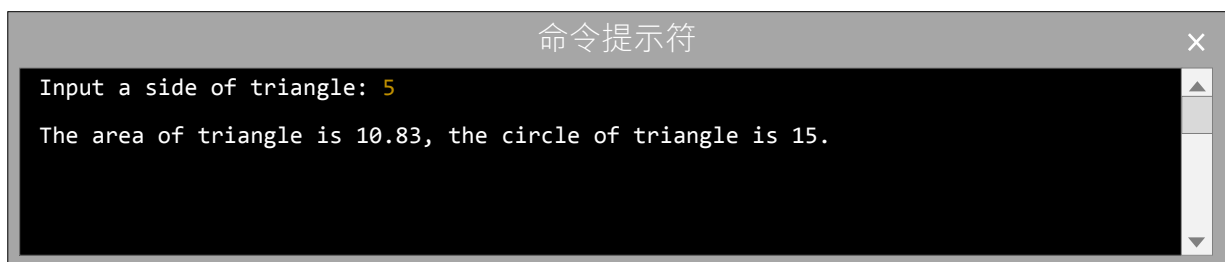


图 735.1

参考答案:

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

```
#include <math.h>

int main(void)
{
    int num0, c0;
    float p0, s0;

    printf("Input a side of triangle: ");
    scanf("%d", &num0);
    p0 = (num0 + num0 + num0) / (float)2;
    s0 = (float)(sqrt(p0 * (p0 - num0) * (p0 - num0) * (p0 - num0)));
    c0 = 3 * num0;
    printf("\nThe area of triangle is %.2f, the circle of triangle is %d.\n", s0, c0);

    return 0;
}
```

P736

DIFFICUTLY ★

从键盘上输入一个四位整数，计算各个位上的数字之和。

可用素材: `printf("Input a number with 4-digit: ")`
`printf("\nsum=...\n")...`

程序的运行效果应类似地如图 736.1 所示，金色部分是从键盘输入的内容。



图 736.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int num, qian, bai, shi, ge, sum;

    printf("Input a number with 4-digit: ");
    scanf("%d", &num);
    ge = num % 10;
    shi = num / 10 % 10;
    bai = num / 100 % 10;
    qian = num / 1000;
    sum = ge + shi + bai + qian;
    printf("\nsum=%d\n", sum);

    return 0;
}
```


P111

DIFFICUTLY ★

x (x 只考虑整数 `int` 且必须定义为 `int`, 但 $F(x)$ 完全可能超过 `int` 的表示范围) 通过键盘输入 (输入前给出提示 “Please input x :”) , 然后计算并在屏幕上输出函数值。

$$F(x) = \begin{cases} -5x + 27 & (x < 0) \\ 7909 & (x = 0) \\ 2x - 1 & (x > 0) \end{cases}$$

可用素材: `printf("Please input x: ")`
`printf("\nF(...) = ...`

程序的运行效果应类似地如图 111.1 所示, 金色部分是从键盘输入的内容。



图 111.1

参考答案:

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

```
int main(void)
{
    int x0;
    double F0;

    printf("Please input x: ");
    scanf("%d", &x0);
    if (x0 < 0)
    {
        F0 = (-5) * (double)x0 + 27;
    }
    else if (x0 == 0)
    {
        F0 = 7909;
    }
    else
    {
        F0 = 2 * (double)x0 - 1;
    }
    printf("\nF(%d) = %.01f\n", x0, F0);

    return 0;
}
```

P113

DIFFICUTLY ★

已知某银行整存整取存款不同期限的年息利率分别为: 3.15% 期限一年, 3.63% 期限二年, 4.02% 期

限三年, 4.69% 期限五年, 5.36% 期限八年。从键盘上输入存钱的本金（以元为单位, 必须定义为 int 且应考虑金额很大的情况）和存款期限（只考虑 1、2、3、5、8），求到期时能从银行得到的利息（以元为单位, 应考虑有小数, 不计复利）。

可用素材: `printf("Please input benjin,cunqi: ")`
`printf("\nlixi = ... yuan"`

程序的运行效果应类似地如图 113.1 所示, 金色部分是从键盘输入的内容。



图 113.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    long benjin;
    int cunqi;

    printf("Please input benjin,cunqi: ");
    scanf("%ld,%d", &benjin, &cunqi);
    switch (cunqi)
    {
        case 1:
            printf("\nlixi = %.2f yuan\n", benjin * 0.0315 * 1);
            break;
        case 2:
            printf("\nlixi = %.2f yuan\n", benjin * 0.0363 * 2);
            break;
        case 3:
            printf("\nlixi = %.2f yuan\n", benjin * 0.0402 * 3);
            break;
        case 5:
            printf("\nlixi = %.2f yuan\n", benjin * 0.0469 * 5);
            break;
        case 8:
            printf("\nlixi = %.2f yuan\n", benjin * 0.0536 * 8);
            break;
        default:
            printf("\nInvalid cunqi\n");
    }

    return 0;
}
```

P114

DIFFICUTLY ★

编写一个简单计算器程序, 输入格式为: $data_1 \text{ op } data_2$ 。其中 $data_1$ 和 $data_2$ 是参加运算的两个数 ($data_1$ 、 $data_2$, 必须定义为 int, 但二者相加或相乘可能超出 int 能表示的范围), op 为运算符, 它的取值只能是 “+” “-” “*” “/” “%”。

可用素材: `printf("Please input data1 op data2: ")`
`printf("\nError! chu shu wei 0.\n")`

程序的运行效果应类似地如图 114.1、图 114.2、图 114.3、图 114.4 所示，金色部分是从键盘输入的内容。

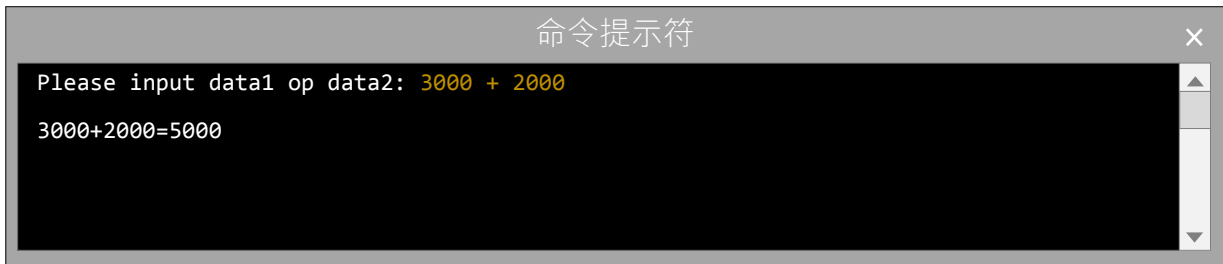


图 114.1 加法



图 114.2 取余



图 114.3 取余时除数为 0

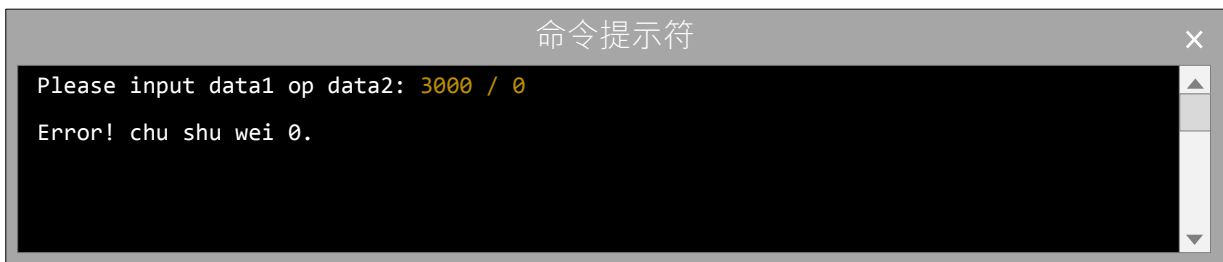


图 114.4 相除时除数为 0

参考答案:

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

```
int main(void)
```

```
{
```

```
    int data1, data2, divide;
```

```
    double data3;
```

```
    char op;
```

```
    printf("Please input data1 op data2: ");
```

```
    scanf("%d %c %d", &data1, &op, &data2);
```

```
    switch (op)
```

```
    {
```

```
        case '+':
```

```
            data3 = (double)data1 + (double)data2;
```

```
            printf("\n%d+%d=%.01f\n", data1, data2, data3);
```

```

        break;
    case '-':
        data3 = (double)data1 - (double)data2;
        printf("\nd-%d=%.0lf\n", data1, data2, data3);
        break;
    case '*':
        data3 = (double)data1 * (double)data2;
        printf("\nd*%d=%d\n", data1, data2, data3);
        break;
    case '/':
        if (0 == data2)
        {
            printf("\nError! chu shu wei 0.\n");
        }
        else
        {
            divide = data1 / data2;
            printf("\nd/%d=%d\n", data1, data2, divide);
        }
        break;
    case '%':
        if (0 == data2)
        {
            printf("\nError! chu shu wei 0.\n");
        }
        else
        {
            divide = data1 % data2;
            printf("\nd%%d=%d\n", data1, data2, divide);
        }
        break;
    default:
        printf("\nInvalid op\n");
}

return 0;
}

```

P124

DIFFICUTLY ★

从键盘读入两个字符 *cBegin* 和 *cEnd*，要求输出 $\geq cBegin$ 且 $\leq cEnd$ 的所有字符。

可用素材： `printf("Please Input two char: ")`
`printf("\nResult: ")`

程序的运行效果应类似地如图 124.1 所示，金色部分是从键盘输入的内容。



图 124.1

参考答案：

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

```

int main(void)
{
    char cBegin, cEnd;
    int i;

    printf("Please Input two char: ");
    cBegin = getchar();
    cEnd = getchar();

    printf("\nResult: ");
    for (i = cBegin; i <= cEnd; i++)
    {
        printf("%c", i);
    }
    putchar('\n');

    return 0;
}

```

P125

DIFFICUTLY ★

从键盘读入两个字符 *cBegin* 和 *cEnd*，要求输出 $\leq cBegin$ 且 $\geq cEnd$ 的所有字符。

可用素材： `printf("Please Input two char: ");`
`printf("\nResult: ");`

程序的运行效果应类似地如图 125.1 所示，金色部分是从键盘输入的内容。



图 125.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    char cBegin, cEnd;
    int i;

    printf("Please Input two char: ");
    cEnd = getchar();
    cBegin = getchar();

    printf("\nResult: ");
    for (i = cEnd; i >= cBegin; i--)
    {
        printf("%c", i);
    }
    putchar('\n');

    return 0;
}

```

P126

DIFFICUTLY ★

从键盘读入一个字符 *cBegin* 和一个数 *iCount*，要求输出 $\geq cBegin$ 的 *iCount* 个字符。

可用素材： `printf("Please Input a char and a number: ")`
`printf("\nResult: ")`

程序的运行效果应类似地如图 126.1 所示，金色部分是从键盘输入的内容。

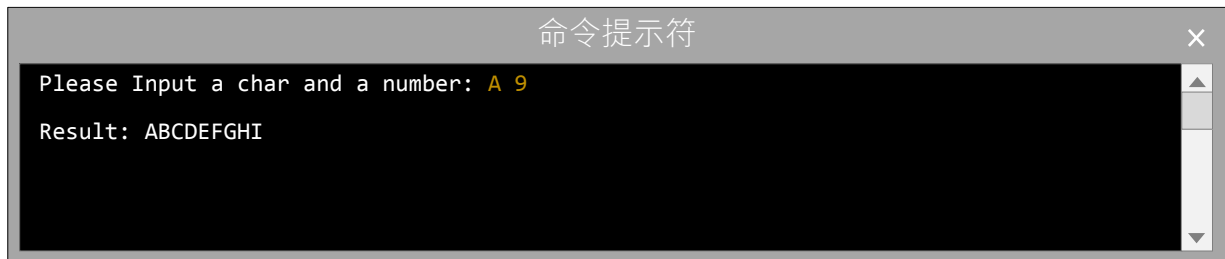


图 126.1

参考答案：

```
#include <stdio.h>

int main(void)
{
    char cBegin;
    int iCount, i;

    printf("Please Input a char and a number: ");
    scanf("%c%d", &cBegin, &iCount);

    printf("\nResult: ");
    for (i = cBegin; i < cBegin + iCount; i++)
    {
        printf("%c", i);
    }
    putchar('\n');

    return 0;
}
```

P127

DIFFICUTLY ★

从键盘读入一个字符 *cBegin* 和一个数 *iCount*，要求输出 $\leq cBegin$ 的 *iCount* 个字符。

可用素材： `printf("Please Input a char and a number: ")`
`printf("\nResult: ")`

程序的运行效果应类似地如图 127.1 所示，金色部分是从键盘输入的内容。



图 127.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    char cBegin;
    int iCount, i;

    printf("Please Input a char and a number: ");
    scanf("%c%d", &cBegin, &iCount);

    printf("\nResult: ");
    for (i = cBegin; i > cBegin - iCount; i--)
    {
        printf("%c", i);
    }
    putchar('\n');

    return 0;
}
```

P238

DIFFICUTLY ★

先从键盘读入 5 个整数，然后倒序输出这 5 个数。

可用素材: `printf("请输入5个数: ")`
`printf("\n这5个数倒序为: ...`

程序的运行效果应类似地如图 238.1 所示，金色部分是从键盘输入的内容。



图 238.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int num1, num2, num3, num4, num5;

    printf("请输入5个数: ");
    scanf("%d%d%d%d%d", &num1, &num2, &num3, &num4, &num5);
    printf("\n这5个数倒序为: %d %d %d %d %d\n", num5, num4, num3, num2, num1);

    return 0;
}
```

P702

DIFFICUTLY ★

输入月份，输出 2003 年该月有几天。当输入的月份超范围时，应输出 “Invalid month input!”。

可用素材: `printf("please input the month number: ")`
`printf("\nInvalid month input !\n")`
`printf("\n2003... has ... days\n")...`

程序的运行效果应类似地如图 702.1 和图 702.2 所示，金色部分是从键盘输入的内容。



图 702.1



图 702.2

参考答案:

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

```
int main(void)
```

```
{
```

```
    int month, big = 31, small = 30, feb = 28;
```

```
    printf("please input the month number: ");
```

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

```
    switch (month)
```

```
    {
```

```
        case 1:
```

```
        case 3:
```

```
        case 5:
```

```
        case 7:
```

```
        case 8:
```

```
        case 10:
```

```
        case 12:
```

```
            printf("\n2003.%d has %d days\n", month, big);
```

```
            break;
```

```
        case 4:
```

```
        case 6:
```

```
        case 9:
```

```
        case 11:
```

```
            printf("\n2003.%d has %d days\n", month, small);
```

```
            break;
```

```
        case 2:
```

```
            printf("\n2003.%d has %d days\n", month, feb);
```

```
            break;
```

```
        default:
```

```
            printf("\nInvalid month input !\n");
```

```
    }
```

```
    return 0;
```

```
}
```


P703

DIFFICUTLY ★

已知某公司员工的保底薪水为 500，某月所接工程的利润 $profit$ （整数）与利润提成的关系如下（计量单位：元）：

$profit \leq 1000$	没有提成；
$1000 < profit \leq 2000$	提成 10%；
$2000 < profit \leq 5000$	提成 15%；
$5000 < profit \leq 10000$	提成 20%；
$10000 < profit$	提成 25%。

请根据输入的利润计算员工的薪水。

可用素材： `printf("Input profit: ")`
`printf("\nsalary=...\n"...`

程序的运行效果应类似地如图 703.1 所示，金色部分是从键盘输入的内容。



图 703.1

参考答案：

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

```
int main(void)
{
    double base = 500, salary;
    int profit;

    printf("Input profit: ");
    scanf("%d", &profit);
    if (profit <= 1000)
    {
        salary = base;
    }
    else if (profit <= 2000)
    {
        salary = base + profit * 0.1;
    }
    else if (profit <= 5000)
    {
        salary = base + profit * 0.15;
    }
    else if (profit <= 10000)
    {
        salary = base + profit * 0.2;
    }
    else
    {
        salary = base + profit * 0.25;
    }
    printf("\nsalary=%.2f\n", salary);
}
```

```
    return 0;
}
```

P713

DIFFICUTLY ★

用 scanf() 输入某年某月某日，判断这一天是这一年的第几天。以 3 月 5 日为例，应该先把前两个月的加起来，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 ≥ 3 时需考虑多加一天。

注：判断年份是否为闰年的方法：为 400 的倍数为闰年，如 2000 年；若非 100 的倍数，而是 4 的倍数，为闰年，如 1996 年。

可用素材： printf("Please input year-month-day: ")
printf("\nIt is the ...th day.\n")...

程序的运行效果应类似地如图 713.1 所示，金色部分是从键盘输入的内容。

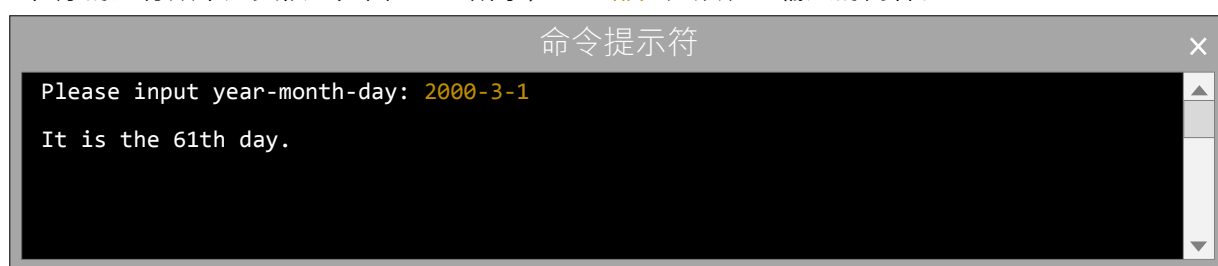


图 713.1

参考答案：

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

```
int main(void)
```

```
{
```

```
    int year, month, day, sum = 0;
```

```
    printf("Please input year-month-day: ");
```

```
    scanf("%d-%d-%d", &year, &month, &day);
```

```
    switch (month - 1)
```

```
    {
```

```
        case 11:
```

```
            sum += 30;
```

```
        case 10:
```

```
            sum += 31;
```

```
        case 9:
```

```
            sum += 30;
```

```
        case 8:
```

```
            sum += 31;
```

```
        case 7:
```

```
            sum += 31;
```

```
        case 6:
```

```
            sum += 30;
```

```
        case 5:
```

```
            sum += 31;
```

```
        case 4:
```

```
            sum += 30;
```

```
        case 3:
```

```
            sum += 31;
```

```
        case 2:
```

```
            sum += 28;
```

```
        case 1:
```

```
            sum += 31;
```

```
        default:
```

```

        ;
    }

    if (year % 4 == 0 && year % 100 != 0 || year % 400 == 0)
    {
        if (month >= 3)
        {
            sum += day + 1;
        }
        else
        {
            sum += day;
        }
    }
    else
    {
        sum += day;
    }
    printf("\nIt is the %dth day.\n", sum);

    return 0;
}

```

P723

DIFFICUTLY ★

输入三角形的三边长 a 、 b 、 c （边长可以是小数），求三角形面积 $area$ ，并输出。如果输入的三边构不成三角形，应给出“data error”的信息提示。

注：根据“海伦—秦九韶”公式， $area = \sqrt{p(p-a)(p-b)(p-c)}$ ，其中 $p = \frac{a+b+c}{2}$ 。

可用素材： `printf("please input triange sides: ")`
`printf("\ndata error\n")`
`printf("\narea=...\n")`

程序的运行效果应类似地如图 723.1 和图 723.2 所示，金色部分是从键盘输入的内容。

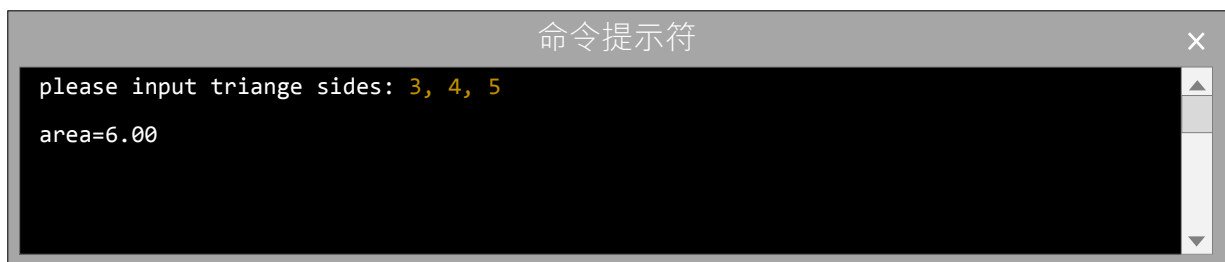


图 723.1



图 723.2

参考答案：

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

```

#include <math.h>

int main(void)
{
    int a0, b0, c0;
    float p0, s0;

    printf("please input triange sides: ");
    scanf("%d,%d,%d", &a0, &b0, &c0);
    if (a0 + b0 > c0 && a0 + c0 > b0 && b0 + c0 > a0)
    {
        p0 = (a0 + b0 + c0) / (float)2;
        s0 = (float)(sqrt(p0 * (p0 - a0) * (p0 - b0) * (p0 - c0)));
        printf("\narea=%.2f\n", s0);
    }
    else
    {
        printf("\ndata error\n");
    }

    return 0;
}

```

P731

DIFFICUTLY ★

输出 n 行星号，每行 5 个星号 “*”。

可用素材： `printf("please input n: ")`

程序的运行效果应类似地如图 731.1 所示，金色部分是从键盘输入的内容。



图 731.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int num, i, star;

    printf("please input n: ");
    scanf("%d", &num);

    putchar('\n');
    for (i = 1; i <= num; i++)
    {
        for (star = 1; star <= 5; star++)
        {
            printf("* ");
        }
        putchar('\n');
    }
}

```

```
    return 0;
}
```

P737

DIFFICUTLY ★

从键盘输入 3 个整数，输出绝对值最大的数。

可用素材： `printf("Input 3 numbers: ")`

`printf("\nThe number with maximum absolute value is ...\n"...`

程序的运行效果应类似地如图 737.1 所示，金色部分是从键盘输入的内容。



图 737.1

参考答案：

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

```
int main(void)
{
    int num1, num2, num3, max;

    printf("Input 3 numbers: ");
    scanf("%d,%d,%d", &num1, &num2, &num3);
    if (abs(num1) >= abs(num2) && abs(num1) >= abs(num3))
    {
        max = num1;
    }
    else if (abs(num2) >= abs(num1) && abs(num2) >= abs(num3))
    {
        max = num2;
    }
    else if (abs(num3) >= abs(num1) && abs(num3) >= abs(num2))
    {
        max = num3;
    }
    printf("\nThe number with maximum absolute value is %d.\n", max);

    return 0;
}
```

P738

DIFFICUTLY ★

从键盘上输入两个实数，计算这两个实数的商（前面的数除以后面的数）。

可用素材： `printf("Input 2 numbers: ")`

`printf("\nThe result is: ...`

`printf("\nDivid by zero")`

程序的运行效果应类似地如图 738.1 和图 738.2 所示，金色部分是从键盘输入的内容。



图 738.1



图 738.2

参考答案:

```
#include <stdio.h>

int main(void)
{
    float num1, num2, num3;

    printf("Input 2 numbers: ");
    scanf("%f%f", &num1, &num2);
    if (num2 == 0)
    {
        printf("\nDivid by zero\n");
    }
    else
    {
        num3 = num1 / num2;
        printf("\nThe result is: %.2f\n", num3);
    }

    return 0;
}
```

P749

DIFFICUTLY ★

用键盘输入的整数产生 5×5 矩阵 **N**，并按行输出该矩阵，每个元素占 4 个数位、右对齐。

可用素材: `printf("Please input an integer: ")`。

程序的运行效果应类似地如图 749.1 和图 749.2 所示，金色部分是从键盘输入的内容。



图 749.1



图 749.2

参考答案:

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

```
int main(void)
{
    int num, i, j, arr[5][5];

    printf("Please input an integer: ");
    scanf("%d", &num);

    for (i = 0; i < 5; i++)
    {
        for (j = 0; j < 5; j++)
        {
            arr[i][j] = num + i + j;
        }
        putchar('\n');
    }
    for (i = 0; i < 5; i++)
    {
        for (j = 0; j < 5; j++)
        {
            printf("%4d", arr[i][j]);
        }
        putchar('\n');
    }

    return 0;
}
```

P754

DIFFICULTLY ★

从键盘输入一个一百分制成绩（无小数），将输入的数据转换成等级“ABCDEFGHIJX”。

90 ~ 100 → A, 80 ~ 89 → B, 70 ~ 79 → C, 60 ~ 69 → D, 50 ~ 59 → E, 40 ~ 49 → F, 30 ~ 39 → G, 20 ~ 29 → H, 10 ~ 19 → I, 0 ~ 9 → J, 其它输入超正常范围分数的则为 X。

可用素材: `printf("please input the score(0-100): ")`
`printf("\nscore=..., grade=...`

程序的运行效果应类似地如图 754.1 所示，金色部分是从键盘输入的内容。



图 754.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int score, shi;
    char mark;

    printf("please input the score(0 - 100): ");
    scanf("%d", &score);
    if (score < 0 || score > 100)
    {
        mark = 'X';
    }
    else
    {
        shi = score / 10;
        switch (shi)
        {
            case 10:
            case 9:
                mark = 'A';
                break;
            case 8:
                mark = 'B';
                break;
            case 7:
                mark = 'C';
                break;
            case 6:
                mark = 'D';
                break;
            case 5:
                mark = 'E';
                break;
            case 4:
                mark = 'F';
                break;
            case 3:
                mark = 'G';
                break;
            case 2:
                mark = 'H';
                break;
            case 1:
                mark = 'I';
                break;
            case 0:
                mark = 'J';
                break;
            default:
                mark = 'X';
        }
    }
}
```



```

    }
}
printf("\nscore=%d, grade=%c\n", score, mark);

return 0;
}

```

P755

DIFFICUTLY ★

从键盘读入一个等级成绩，输出对应的百分制成绩。

A→95、B→85、C→75、D→65、E→55、F→45、G→35、H→25、I→15、J→0。若输入的等级成绩非 A ~ J，则输出 “Error”。

可用素材： `printf("please input the grade: ")`
`printf("\ngrade=..., score=...\n"...`
`printf("\ngrade=... Error\n"...`

程序的运行效果应类似地如图 755.1 和图 755.2 所示，金色部分是从键盘输入的内容。



图 755.1



图 755.2

参考答案：

```

#include <stdio.h>

int main(void)
{
    char grade;
    int score;

    printf("please input the grade: ");
    grade = getchar();
    switch (grade)
    {
        case 'A':
            score = 95;
            printf("\ngrade=%c, score=%d\n", grade, score);
            break;
        case 'B':
            score = 85;
            printf("\ngrade=%c, score=%d\n", grade, score);
            break;

```

```

    case 'C':
        score = 75;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'D':
        score = 65;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'E':
        score = 55;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'F':
        score = 45;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'G':
        score = 35;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'H':
        score = 25;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'I':
        score = 15;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    case 'J':
        score = 0;
        printf("\ngrade=%c, score=%d\n", grade, score);
        break;
    default:
        printf("\ngrade=%c Error\n", grade);
}

return 0;
}

```

P778

DIFFICUTLY ★

字符串“abcd”每个字符都向右移位，最右的则移动到第一个字符的位置，就变为“dabc”，这称为对串进行位移为 1 的轮换。同理，“abcd”变为“cdab”则称为位移为 2 的轮换。要求从键盘读入一个字符串 *str*（约定字符串中字符数 ≤ 80 字节，字符串中可以有空格）和需要位移的值 *n*（ $n > str$ 的长度时，循环位移），输出对该字符串进行位移为 *n* 的轮换结果。

可用素材： printf("Please input the string: ")
printf("Please input n: ")
printf("\nThe result is: %s\n"...

程序的运行效果应类似地如图 1 和图 2 所示，金色部分是从键盘输入的内容。



图 778.1



图 778.2

参考答案:

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

int main(void)
{
    char str[81];
    int num, len, i;

    printf("Please input the string: ");
    gets(str);
    len = strlen(str);
    printf("Please input n: ");
    scanf("%d", &num);

    printf("\nThe result is: ");
    if (num <= len)
    {
        for (i = len - num; i < len; i++)
        {
            printf("%c", str[i]);
        }
        for (i = 0; i < len - num; i++)
        {
            printf("%c", str[i]);
        }
    }
    else
    {
        for (i = len - num % len; i < len; i++)
        {
            printf("%c", str[i]);
        }
        for (i = 0; i < len - num % len; i++)
        {
            printf("%c", str[i]);
        }
    }
    putchar('\n');

    return 0;
}
```

```
}
```

P827

DIFFICUTLY ★

从键盘输入 3 个可带空格的字符串（约定：字符数 ≤ 127 字节），输出长度最大的字符串的长度。

可用素材：

```
printf("Please input the first string:\t")
printf("Please input the second string:\t")
printf("Please input the third string:\t")
printf("\n最长的字符串长度为： ...
```

程序的运行效果应类似地如图 827.1 所示，金色部分是从键盘输入的内容。



图 827.1

参考答案：

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

int main(void)
{
    char str1[128], str2[128], str3[128];
    int len1, len2, len3, max;

    printf("Please input the first string:\t");
    gets(str1);
    len1 = strlen(str1);
    printf("Please input the second string:\t");
    gets(str2);
    len2 = strlen(str2);
    printf("Please input the third string:\t");
    gets(str3);
    len3 = strlen(str3);

    if (len1 > len2)
    {
        max = len1;
    }
    else
    {
        max = len2;
    }
    if (len3 > max)
    {
        max = len3;
    }
    printf("\n最长的字符串长度为： %d\n", max);

    return 0;
}
```

P828

DIFFICUTLY ★

从键盘输入 3 个可带空格的字符串（约定：字符数 ≤ 127 字节），输出最大的字符串。

可用素材： `printf("Please input the first string:\t")`
`printf("Please input the second string:\t")`
`printf("Please input the third string:\t")`
`printf("\n最大字符串是： ...`

程序的运行效果应类似地如图 828.1 所示，金色部分是从键盘输入的内容。

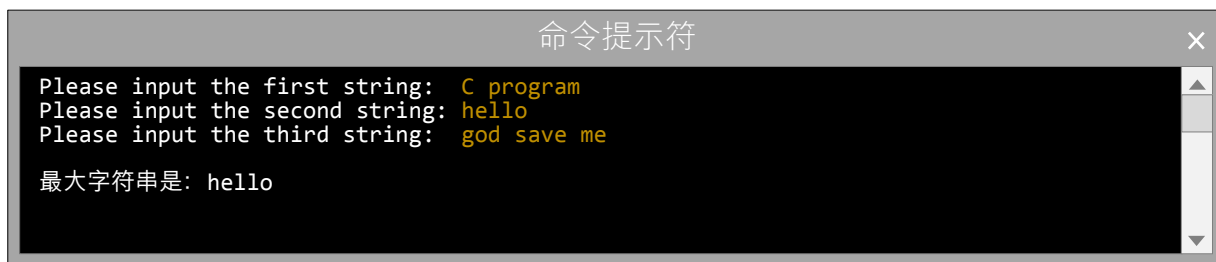


图 828.1

参考答案：

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

int main(void)
{
    char str1[128], str2[128], str3[128], strmax[128];

    printf("Please input the first string:\t");
    gets(str1);
    printf("Please input the second string:\t");
    gets(str2);
    printf("Please input the third string:\t");
    gets(str3);

    if (strcmp(str1, str2) > 0)
    {
        strcpy(strmax, str1);
    }
    else
    {
        strcpy(strmax, str2);
    }
    if (strcmp(str3, strmax) > 0)
    {
        strcpy(strmax, str3);
    }
    printf("\n最大字符串是: %s\n", strmax);

    return 0;
}
```

P112

DIFFICUTLY ★

设某企业 2006 年的产值为 5000 万，计划以后每年的增长率为 x (x 从键盘输入，例如输入 8.75 表示 8.75%)，计算该企业的产值在哪年实现翻番以及翻番时的产值，然后输出（输出时以万为单位，应考虑有小数）。

可用素材: `printf("Please input x: ")`
`printf("\nyear = ... nian, chanzhi = ...`

程序的运行效果应类似地如图 112.1 所示, 金色部分是从键盘输入的内容。



图 112.1

参考答案:

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

```
int main(void)
{
    double increase, output = 5000;
    int year;

    printf("Please input x: ");
    scanf("%lf", &increase);

    for (year = 2007; ; year++)
    {
        output += output * increase / 100;
        if (output >= 10000)
        {
            break;
        }
    }
    printf("\nyear = %d nian, chanzhi = %.2f\n", year, output);

    return 0;
}
```

P115

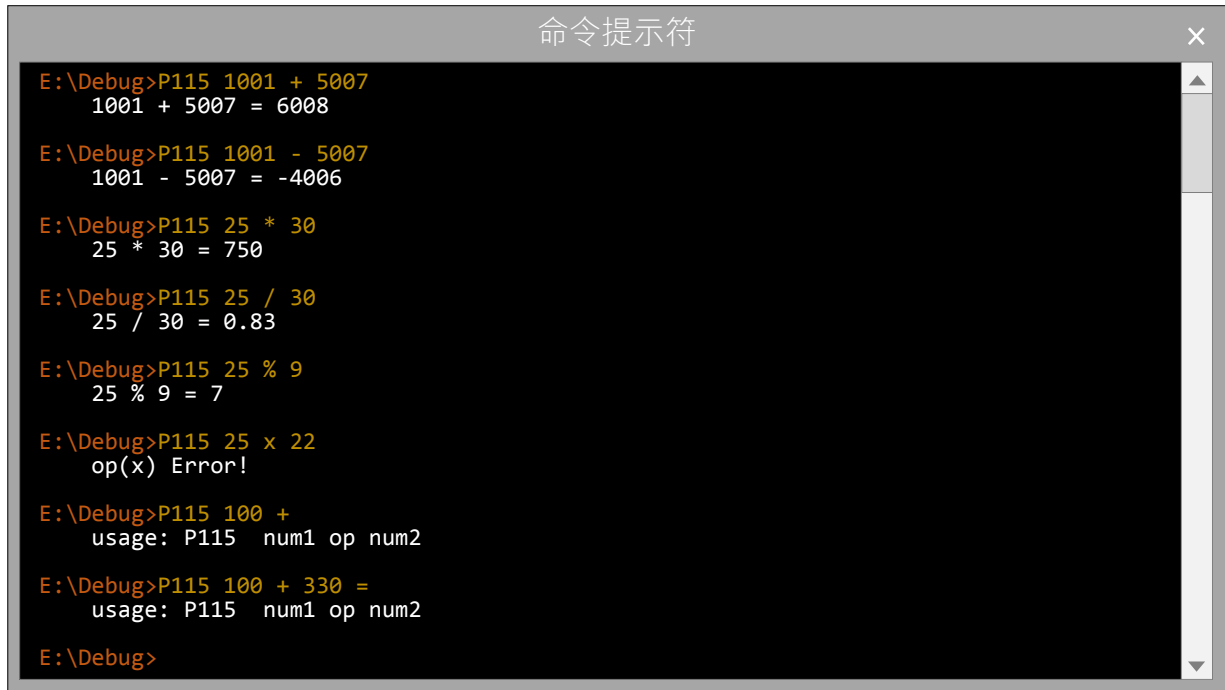
DIFFICUTLY ★

制作一简单的计算器。注意事项:

1. 需要计算的内容从命令行输入, 格式为: `P115 num1 op num2`, 当命令行格式不正确 (参数个数不为 4) 时, 应报错。
2. `op` 的取值范围为 “+” “-” “*” “/” “%”, 超出此范围则应报错。
3. `num1` 和 `num2` 均为整数 (int), `op` 为 “+” “-” “*” 时不考虑运算结果超出 int 型能表示的范围, `op` 为 “/” “%” 时不考虑除数为 0 的情况, 但 `op` 为 “/” 时计算结果应保留 2 位小数。
4. 程序的返回值 (即由 `main()` 函数 `return` 的值和程序使用 `exit()` 终止运行时返回的值, 也称退出代码) 规定为:
 - a) 正常运行结束时, 返回 0。
 - b) 命令行格式不对, 返回 1。
 - c) `op` 超出范围时, 返回 2。

可用素材: `printf(" usage: P115 num1 op num2\n")`
`printf(" op(...) Error!\n")...`

程序的运行效果应类似地如图 115.1 所示，**橙色部分**为系统命令行提示符，表示程序 P115.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



```
E:\Debug>P115 1001 + 5007
1001 + 5007 = 6008

E:\Debug>P115 1001 - 5007
1001 - 5007 = -4006

E:\Debug>P115 25 * 30
25 * 30 = 750

E:\Debug>P115 25 / 30
25 / 30 = 0.83

E:\Debug>P115 25 % 9
25 % 9 = 7

E:\Debug>P115 25 x 22
op(x) Error!

E:\Debug>P115 100 +
usage: P115 num1 op num2

E:\Debug>P115 100 + 330 =
usage: P115 num1 op num2

E:\Debug>
```

图 115.1

参考答案:

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

int main(int argc, char *argv[])
{
    int num1, num2;

    if (argc != 4)
    {
        printf("    usage: P115 num1 op num2\n");
        return 1;    /* 可用 exit(1), 但需要标头 stdlib.h */
    }

    num1 = atoi(argv[1]);
    num2 = atoi(argv[3]);
    switch (argv[2][0])
    {
        case '+':
            printf("    %d + %d = %d\n", num1, num2, num1 + num2);
            break;
        case '-':
            printf("    %d - %d = %d\n", num1, num2, num1 - num2);
            break;
        case '*':
            printf("    %d * %d = %d\n", num1, num2, num1 * num2);
            break;
        case '/':
            printf("    %d / %d = %.2f\n", num1, num2, (double)num1 / num2);
            break;
        case '%':
            printf("    %d %% %d = %d\n", num1, num2, num1 % num2);
            break;
        default:
            break;
    }
}
```

```

        printf("    op(%c) Error!\n", argv[2][0]);
        return 2;    /* 可用 exit(2), 但需要标头 stdlib.h, 以下程序同 */
    }

    return 0;
}

```

P116

DIFFICUTLY ★

从命令行输入两个实数，格式为：P116 num_1 num_2 ，输出 $(num_1 + num_2) \div 2$ 之值且保留 3 位小数。
提示与注意事项：

1. 库函数提示：atof()。
2. 当命令行格式不正确（参数个数不为 3）时，应报错。
3. 程序的返回值（即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值，也称退出代码）规定为：
 - a) 正常运行结束时，返回 0。
 - b) 命令行格式不对，返回 9。

可用素材： printf(" usage: P116 num1 num2\n")
printf(" (... + ...) / 2 = ...\n")...

程序的运行效果应类似地如图 116.1 所示，**橙色部分**为系统命令行提示符，表示程序 P116.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



图 116.1

参考答案：

```

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

int main(int argc, char *argv[])
{
    double num1, num2;

    if (argc != 3)
    {
        printf("    usage: P116 num1 num2\n");
        return 9;
    }

    num1 = atof(argv[1]);
    num2 = atof(argv[2]);
    printf("    (%.3f + %.3f) / 2 = %.3f\n", num1, num2, (num1 + num2) / 2);
    return 0;
}

```


P117

DIFFICUTLY ★

从命令行输入两个实数，格式为：P117 num_1 num_2 ，输出 $(num_1 - num_2) \times 3$ 之值且保留 3 位小数。
提示与注意事项：

1. 库函数提示：atof。
2. 当命令行格式不正确（参数个数不为 3）时，应报错。
3. 程序的返回值（即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值，也称退出代码）规定为：
 - a) 正常运行结束时，返回 0。
 - b) 命令行格式不对，返回 76。

可用素材： printf(" usage: P117 num1 num2\n")
printf(" (... - ...) * 3 = ...\n"...

程序的运行效果应类似地如图 1 所示，**橙色部分**为命令行提示符，表示程序 P117.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



图 117.1

参考答案：

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

int main(int argc, char *argv[])
{
    double num1, num2;

    if (argc != 3)
    {
        printf(" usage: P117 num1 num2\n");
        return 76;
    }

    num1 = atof(argv[1]);
    num2 = atof(argv[2]);
    printf(" (%.3f - %.3f) * 3 = %.3f\n", num1, num2, (num1 - num2) * 3);
    return 0;
}
```

P118

DIFFICUTLY ★

从命令行输入两个实数，格式为：P118 num_1 num_2 ，输出 $(num_1^2 - num_2^2) \div 6$ 之值且保留 3 位小数。
提示与注意事项：

1. 库函数提示：atof。
2. 当命令行格式不正确（参数个数不为 3）时，应报错。

3. 程序的返回值 (即由 `main()` 函数 `return` 的值和程序使用 `exit()` 终止运行时返回的值, 也称退出代码) 规定为:
- a) 正常运行结束时, 返回 0。
 - b) 命令行格式不对, 返回 103。

可用素材: `printf(" usage: P118 num1 num2\n")`
`printf(" (...*... - ...*...) / 6 = ...\n"...`

程序的运行效果应类似地如图 118.1 所示, 橙色部分为命令行提示符, 表示程序 P118.exe 所在的文件夹, 用户的程序位置可不必如此; 金色部分是从命令行输入的内容。

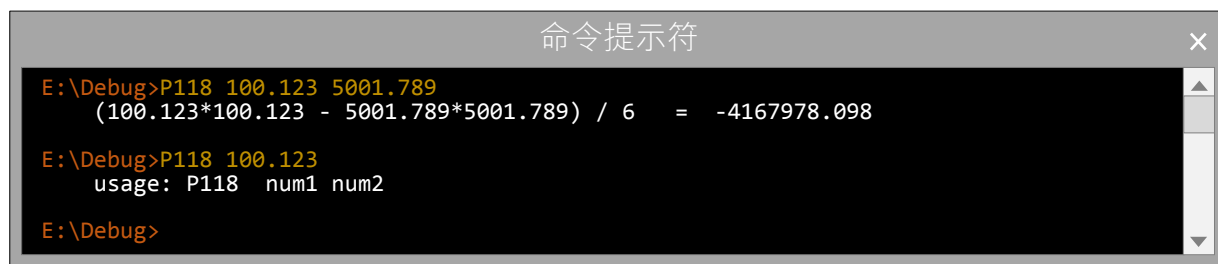


图 118.1

参考答案:

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

```
int main(int argc, char *argv[])
{
    double num1, num2;

    if (argc != 3)
    {
        printf("    usage: P118 num1 num2\n");
        return 103;
    }

    num1 = atof(argv[1]);
    num2 = atof(argv[2]);
    printf("    (%.3f*%.3f - %.3f*%.3f) / 6    = %.3f\n",
        num1, num1, num2, num2, (num1 * num1 - num2 * num2) / 6);
    return 0;
}
```

P119

DIFFICUTLY ★

从命令行输入三个数, 格式为: P119 num_1 num_2 num_3 , 输出 $num_1 + (num_2 + num_3) \div 2$ 之值且保留 3 位小数。提示与注意事项:

1. 库函数提示: `atoi()`、`atof()`。
2. 当命令行格式不正确 (参数个数不为 4) 时, 应报错。
3. 程序的返回值 (即由 `main()` 函数 `return` 的值和程序使用 `exit()` 终止运行时返回的值, 也称退出代码) 规定为:
 - a) 正常运行结束时, 返回 0。
 - b) 命令行格式不对, 返回 8。

可用素材: `printf(" usage: P119 num1 num2 num3\n")`
`printf(" ... + (... + ...) / 2 = ...\n"...`

程序的运行效果应类似地如图 119.1 所示，**橙色部分**为命令行提示符，表示程序 P119.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



图 119.1

参考答案:

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

int main(int argc, char *argv[])
{
    int num1;
    double num2, num3;

    if (argc != 4)
    {
        printf("    usage: P119 num1 num2 num3\n");
        return 8;
    }

    num1 = atoi(argv[1]);
    num2 = atof(argv[2]);
    num3 = atof(argv[3]);
    printf("    %d + (%.3f + %.3f) / 2 = %.3f\n", num1, num2, num3, num1 + (num2 + num3) / 2);
    return 0;
}
```

P120

DIFFICUTLY ★

从命令行输入三个数，格式为：P120 num_1 num_2 num_3 ，输出 $num_1 + (num_2 - num_3) \times 3$ 之值且保留 3 位小数。提示与注意事项：

1. 库函数提示：atoi()、atof()。
2. 当命令行格式不正确（参数个数不为 4）时，应报错。
3. 程序的返回值（即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值，也称退出代码）规定为：
 - a) 正常运行结束时，返回 0。
 - b) 命令行格式不对，返回 23。

可用素材： printf(" usage: P120 num1 num2 num3\n")
printf(" ... + (... - ...) * 3 = ...\n")...

程序的运行效果应类似地如图 120.1 所示，**橙色部分**为命令行提示符，表示程序 P120.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



图 120.1

参考答案:

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

```
int main(int argc, char *argv[])
{
    int num1;
    double num2, num3;

    if (argc != 4)
    {
        printf("    usage: P120 num1 num2 num3\n");
        return 23;
    }

    num1 = atoi(argv[1]);
    num2 = atof(argv[2]);
    num3 = atof(argv[3]);
    printf("    %d + (%.3f - %.3f) * 3 = %.3f\n", num1, num2, num3, num1 + (num2 - num3) * 3);
    return 0;
}
```

P121

DIFFICUTLY ★

从命令行输入三个数，格式为：P121 num_1 num_2 num_3 ，输出 $num_1 + (num_2^2 - num_3^2) \div 6$ 之值且保留 3 位小数。提示与注意事项：

1. 库函数提示：atoi()、atof()。
2. 当命令行格式不正确（参数个数不为 4）时，应报错。
3. 程序的返回值（即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值，也称退出代码）规定为：
 - a) 正常运行结束时，返回 0。
 - b) 命令行格式不对，返回 97。

可用素材： printf(" usage: P121 num1 num2 num3\n")
printf(" ... + (...*... - ...*...) / 6 = ...\n"...

程序的运行效果应类似地如图 121.1 所示，**橙色部分**为命令行提示符，表示程序 P121.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



图 121.1

参考答案:

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

int main(int argc, char *argv[])
{
    int num1;
    double num2, num3, result;

    if (argc != 4)
    {
        printf("    usage: P121  num1 num2 num3\n");
        return 97;
    }

    num1 = atoi(argv[1]);
    num2 = atof(argv[2]);
    num3 = atof(argv[3]);
    result = num1 + (num2 * num2 - num3 * num3) / 6;
    printf("    %d + (%.3f*%.3f - %.3f*%.3f) / 6 = %.3f\n", num1, num2, num2, num3, num3, result);
    return 0;
}
```

P221

DIFFICUTLY ★

从键盘读入一个字符串（约定：字符数 ≤ 127 字节），检查该字符串是否是回文。所谓回文即正向与反向的拼写都一样，例如“adgda”。

可用素材： `printf("Please input string: ")`
`printf("\n... shi hui wen."...`
`printf("\n... bu shi hui wen."...`

程序的运行效果应类似地如图 221.1 和图 221.2 所示，金色部分是从键盘输入的内容。



图 221.1



图 221.2

参考答案:

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

int main(void)
{
    char str[128];
    int len, i, j, flag = 0;

    printf("Please input string: ");
    gets(str);
    len = strlen(str);

    for (i = 0, j = len - 1; i <= j; i++, j--)
    {
        if (str[i] != str[j])
        {
            flag = 1;
            break;
        }
    }
    if (flag == 0)
    {
        printf("\n%s shi hui wen.\n", str);
    }
    else
    {
        printf("\n%s bu shi hui wen.\n", str);
    }

    return 0;
}
```

P224

DIFFICUTLY ★

猴子吃桃问题。猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个。第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第 n (n 从键盘输入) 天早上想再吃时，就只剩下一个桃子了。求第一天共摘了多少个桃子（不考虑猴子是否真的能吃多少桃子）。

可用素材: `printf("Please input n: ")`
`printf("\ntotal=..."`

程序的运行效果应类似地如图 224.1 所示，金色部分是从键盘输入的内容。



图 224.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int day, total = 1, i;

    printf("Please input n: ");
    scanf("%d", &day);
    for (i = day - 1; i >= 1; i--)
    {
        total = (total + 1) * 2;
    }
    printf("\ntotal=%d\n", total);

    return 0;
}
```

P225

DIFFICULTLY ★

从键盘读入一个整数 *Num*，按从小到大的顺序依次输出所有满足条件的 3 位数：该数各位数字的立方和等于 *Num*。

可用素材:

```
printf("Please Input a number: ")
printf("\nResult: ")
printf("%5d"...
printf("not Find!\n")
```

程序的运行效果应类似地如图 225.1 和图 225.2 所示，金色部分是从键盘输入的内容。

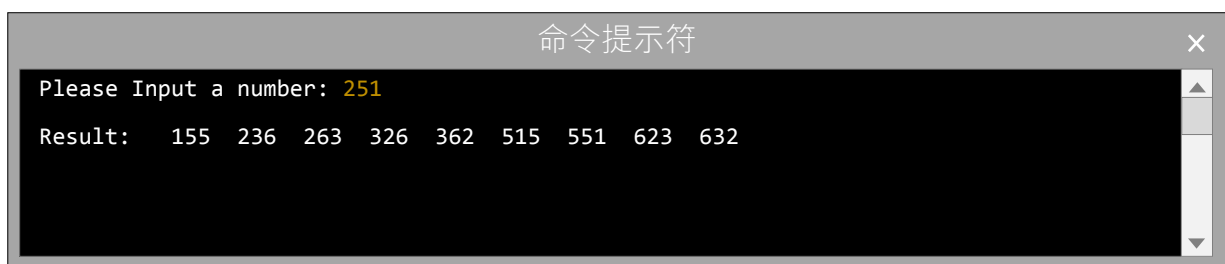


图 225.1 所有各位数字之立方和等于 251 的 3 位数



图 225.2 未找到各位数字之立方和等于 300 的 3 位数

参考答案:

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

```
int main(void)
{
    int num1, num2, ge, shi, bai, i, flag = 0;

    printf("Please Input a number: ");
    scanf("%d", &num1);

    printf("\nResult: ");
    for (i = 100; i < 1000; i++)
    {
        ge = i % 10;
        shi = i / 10 % 10;
        bai = i / 100;
        num2 = bai * bai * bai + shi * shi * shi + ge * ge * ge;
        if (num1 == num2)
        {
            printf("%5d", i);
            flag = 1;
        }
    }
    if (flag == 0)
    {
        printf("not Find!");
    }
    putchar('\n');

    return 0;
}
```

P226

DIFFICUTLY ★

从键盘读入两个整数 $iBegin$ 和 $iEnd$, 要求输出 $\geq iBegin$ 且 $\leq iEnd$ 的所有整数。

可用素材: `printf("Please Input two number: ")`
`printf("\nResult: ")`
`printf(" %d"...`

程序的运行效果应类似地如图 226.1 所示, 金色部分是从键盘输入的内容。

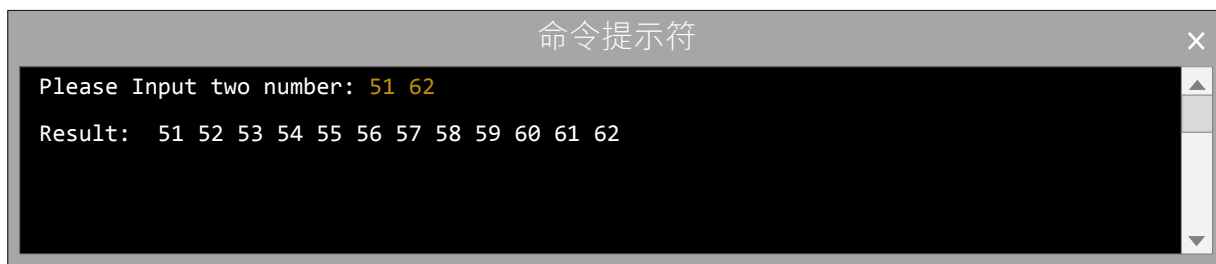


图 226.1

参考答案:

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

```
int main(void)
{
    int num1, num2, iBegin, iEnd, i;

    printf("Please Input two number: ");
    scanf("%d%d", &num1, &num2);
    if (num2 > num1)
    {
        iBegin = num1;
        iEnd = num2;
    }
    else
    {
        iBegin = num2;
        iEnd = num1;
    }

    printf("\nResult: ");
    for (i = iBegin; i <= iEnd; i++)
    {
        printf(" %d", i);
    }
    putchar('\n');

    return 0;
}
```

P227

DIFFICUTLY ★

从键盘读入两个整数 $iBegin$ 和 $iEnd$ ，要求输出 $\leq iBegin$ 且 $\geq iEnd$ 的所有整数。

可用素材: `printf("Please Input two number: ")`
`printf("\nResult: ")`
`printf(" %d"...`

程序的运行效果应类似地如图 1 所示，金色部分是从键盘输入的内容。

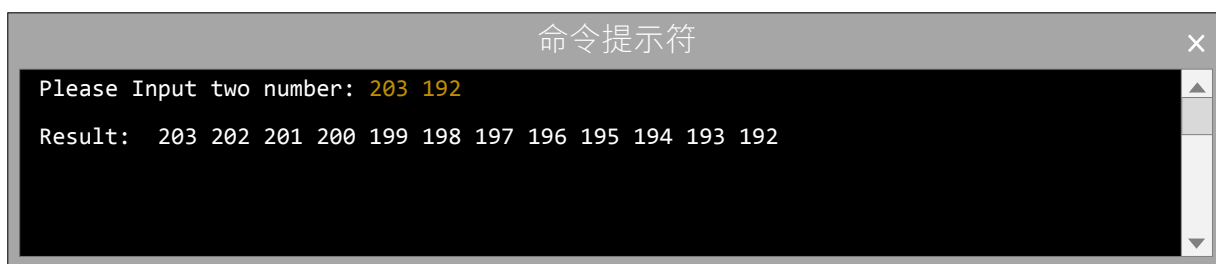


图 227.1

参考答案:

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

```
int main(void)
{
    int num1, num2, iBegin, iEnd, i;

    printf("Please Input two number: ");
    scanf("%d%d", &num1, &num2);
    if (num2 > num1)
    {
        iBegin = num2;
        iEnd = num1;
    }
    else
    {
        iBegin = num1;
        iEnd = num2;
    }

    printf("\nResult: ");
    for (i = iBegin; i >= iEnd; i--)
    {
        printf(" %d", i);
    }
    putchar('\n');

    return 0;
}
```

P228

DIFFICUTLY ★

从键盘读入两个整数 *iBegin* 和 *iCount*, 要求输出 $\geq iBegin$ 的 *iCount* 个整数

可用素材: `printf("Please Input two number: ")`
`printf("\nResult: ")`
`printf(" %d"...`

程序的运行效果应类似地如图 228.1 所示, 金色部分是从键盘输入的内容。



图 228.1

参考答案:

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

```
int main(void)
{
    int iBegin, iCount, i;

    printf("Please Input two number: ");
    scanf("%d%d", &iBegin, &iCount);
```

```

printf("\nResult: ");
for (i = 1; i <= iCount; i++)
{
    printf(" %d", iBegin);
    iBegin++;
}
putchar('\n');

return 0;
}

```

P229

DIFFICUTLY ★

从键盘读入两个整数 *iBegin* 和 *iCount*，要求输出 $\leq iBegin$ 的 *iCount* 个整数

可用素材： `printf("Please Input two number: ")`
`printf("\nResult: ")`
`printf(" %d"...`

程序的运行效果应类似地如图 229.1 所示，金色部分是从键盘输入的内容。



图 229.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int iBegin, iCount, i;

    printf("Please Input two number: ");
    scanf("%d%d", &iBegin, &iCount);

    printf("\nResult: ");
    for (i = 1; i <= iCount; i++)
    {
        printf(" %d", iBegin);
        iBegin--;
    }
    putchar('\n');

    return 0;
}

```

P230

DIFFICUTLY ★

从键盘读入两个整数 *iBegin* 和 *iCount*，要求输出 $\geq iBegin$ 的 *iCount* 个整数（后一数为前一数加 5）。

可用素材： `printf("Please Input two number: ")`

```
printf("\nResult: ")
printf(" %d"...
```

程序的运行效果应类似地如图 230.1 所示，金色部分是从键盘输入的内容。



图 230.1

参考答案:

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

```
int main(void)
{
    int iBegin, iCount, i;

    printf("Please Input two number: ");
    scanf("%d%d", &iBegin, &iCount);

    printf("\nResult: ");
    for (i = 1; i <= iCount; i++)
    {
        printf(" %d", iBegin);
        iBegin += 5;
    }
    putchar('\n');

    return 0;
}
```

P231

DIFFICUTLY ★

从键盘读入两个整数 *iBegin* 和 *iCount*，要求输出 $\leq iBegin$ 的 *iCount* 个整数（后一数为前一数减 7）。

可用素材: `printf("Please Input two number: ")`
`printf("\nResult: ")`
`printf(" %d"...`

程序的运行效果应类似地如图 231.1 所示，金色部分是从键盘输入的内容。



图 231.1

参考答案:

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

```
int main(void)
```

```

{
    int iBegin, iCount, i;

    printf("Please Input two number: ");
    scanf("%d%d", &iBegin, &iCount);

    printf("\nResult: ");
    for (i = 1; i <= iCount; i++)
    {
        printf(" %d", iBegin);
        iBegin -= 7;
    }
    putchar('\n');

    return 0;
}

```

P232

DIFFICUTLY ★

从键盘读入一个整数 *Num*，按从小到大的顺序依次输出所有满足条件的 3 位数：该数各位数字的平方和再加上该数除以 2 的值等于 *Num*。

可用素材：

```

printf("Please Input a number: ")
printf("\nResult: ")
printf("%5d"...
printf("not Find!\n")

```

程序的运行效果应类似地如图 232.1 和图 232.2 所示，金色部分是从键盘输入的内容。



图 232.1



图 232.2

参考答案：

```

#include <stdio.h>

int main(void)
{
    int num1, num2, ge, shi, bai, i, flag = 0;

    printf("Please Input a number : ");
    scanf("%d", &num1);

```

```

printf("\nResult: ");
for (i = 100; i < 1000; i++)
{
    ge = i % 10;
    shi = i / 10 % 10;
    bai = i / 100;
    num2 = bai * bai + shi * shi + ge * ge + i / 2;
    if (num1 == num2)
    {
        printf("%5d", i);
        flag = 1;
    }
}
if (flag == 0)
{
    printf("not Find!");
}
putchar('\n');

return 0;
}

```

P233

DIFFICUTLY ★

从键盘读入一个整数 Num ，按从小到大的顺序依次输出所有满足条件的 3 位数：该数各位数字之积加上该数十位数的平方再加上该数个位数的立方之和等于 Num 。

可用素材： `printf("Please Input a number: ")`
`printf("\nResult: ")`
`printf("%4d"...`
`printf("not Find!\n")`

程序的运行效果应类似地如图 233.1 和图 233.2 所示，金色部分是从键盘输入的内容。



图 233.1



图 233.2

参考答案：

```

#include <stdio.h>

int main(void)
{

```

```

int num1, num2, ge, shi, bai, i, flag = 0;

printf("Please Input a number: ");
scanf("%d", &num1);

printf("\nResult: ");
for (i = 100; i < 1000; i++)
{
    ge = i % 10;
    shi = i / 10 % 10;
    bai = i / 100;
    num2 = bai * shi * ge + shi * shi + ge * ge * ge;
    if (num1 == num2)
    {
        printf("%4d", i);
        flag = 1;
    }
}
if (flag == 0)
{
    printf("not Find!");
}
putchar('\n');

return 0;
}

```

P320

DIFFICUTLY ★

在文本文件“Comp.txt”里有需要计算结果的整数算式，每个算式占一行且文件中只有一个算式，运算类型只有“加法(+)”或者“减法(-)”且运算符前后至少有一个空格。计算该算式的结果并在屏幕上显示。

把程序运行时测试用的算式文件“Comp.txt”（见附件及文件内容）（加法示例，编程时还应考虑算式为减法的情况）保存到程序 P320.c 所在的文件夹且文件名保持不变。

可用素材： `printf("%d + %d = %d\n"...`
`printf("%d - %d = %d\n"...`

程序的运行效果应类似地如图 320.1 和图 320.2 所示。



图 320.1 文件“Comp.txt”中的内容为“123 + 556”



图 320.2 文件“Comp.txt”中的内容为“123 - 556”

附件:  Comp.txt

文件内容:

Comp.txt

123 + 556

参考答案:

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

```
int main(void)
{
    FILE *fp;
    int num1, num2;
    char op;

    fp = fopen("Comp.txt", "r");
    if (fp == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    fscanf(fp, "%d %c %d", &num1, &op, &num2);
    if (op == '+')
    {
        printf("%d + %d = %d\n", num1, num2, num1 + num2);
    }
    else if (op == '-')
    {
        printf("%d - %d = %d\n", num1, num2, num1 - num2);
    }

    fclose(fp);

    return 0;
}
```

P323

DIFFICULTLY ★

在文本文件“Comp.txt”里有需要计算结果的整数算式，每个算式占一行且文件中只有一个算式，运算类型只有“加法(+)”“减法(-)”“乘法(*)”且运算符前后至少有一个空格。计算该算式的结果并在屏幕上显示。

把程序运行时测试用的算式文件“Comp.txt”（见附件及文件内容）保存到程序 P323.c 所在的文件夹且文件名保持不变。

可用素材: `printf("%d %c %d %c %d = %d\n"...`

程序的运行效果应类似地如图 323.1 和图 323.2 所示。

命令提示符

123 + 556 * 2 = 1235

图 323.1 文件 “Comp.txt” 内容为 “123 + 556 * 2”

命令提示符

556 * 2 - 1235 = -123

图 323.2 文件 “Comp.txt” 内容为 “556 * 2 - 1235”

附件:  Comp.txt

文件内容:

Comp.txt

123 + 556 * 2

参考答案:

```
#include <stdio.h>

int Fn(int num1, char op, int num2);

int main(void)
{
    FILE *fp;
    int num1, num2, num3, result;
    char op1, op2;

    fp = fopen("Comp.txt", "r");
    if (fp == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    fscanf(fp, "%d %c %d %c %d", &num1, &op1, &num2, &op2, &num3);
    if (op2 == '*')
    {
        result = Fn(num1, op1, Fn(num2, op2, num3));
    }
    else
    {
        result = Fn(Fn(num1, op1, num2), op2, num3);
    }
    printf("%d %c %d %c %d = %d\n", num1, op1, num2, op2, num3, result);

    fclose(fp);

    return 0;
}
```

```

}

int Fn(int num1, char op, int num2)
{
    int result;

    if (op == '+')
    {
        result = num1 + num2;
    }
    else if (op == '-')
    {
        result = num1 - num2;
    }
    else if (op == '*')
    {
        result = num1 * num2;
    }

    return result;
}

```

P714

DIFFICUTLY ★

用 scanf 输入 10 个整数（采用 int 数据类型），计算所有正数的和、负数的和以及 10 个数的和。

可用素材： printf("Input 10 integers: ")
printf("\nzhengshu=...,fushu=...,all=...")

程序的运行效果应类似地如图 1 所示，金色部分是从键盘输入的内容。

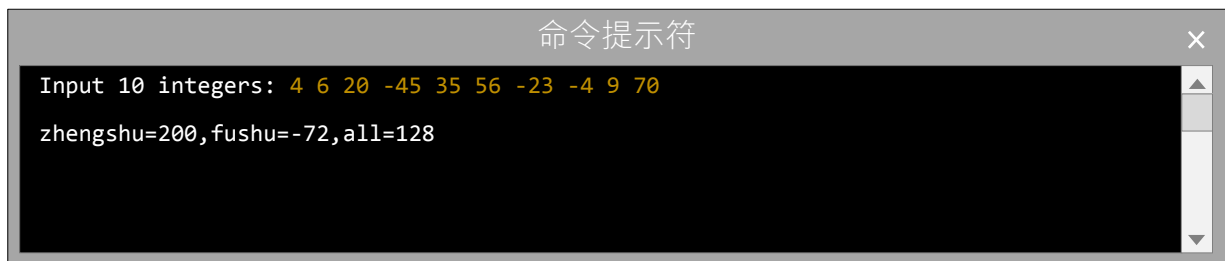


图 714.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int arr[10], i, zhengshu = 0, fushu = 0, all = 0;

    printf("Input 10 integers: ");
    for (i = 0; i < 10; i++)
    {
        scanf("%d", &arr[i]);
        all += arr[i];
        if (arr[i] > 0)
        {
            zhengshu += arr[i];
        }
        else
        {
            fushu += arr[i];
        }
    }
}

```

```

    }
}
printf("\nzhengshu=%d,fushu=%d,all=%d\n", zhengshu, fushu, all);

return 0;
}

```

P718

DIFFICUTLY ★

有一递推数列，满足 $f(0) = 0$, $f(1) = 1$, $f(2) = 2$, $f(n+1) = 2f(n) + f(n-1)f(n-2)$ ($n \geq 2$)，编写程序求 $f(n)$ 的值 (n 由键盘输入, $13 \geq n \geq 2$)。

可用素材: `printf("Input n (13>=n>=2): ")`
`printf("\nf(...)=...\n"...`

程序的运行效果应类似地如图 718.1 所示，金色部分是从键盘输入的内容。



图 718.1

参考答案:

```

#include <stdio.h>

int main(void)
{
    int num, i;
    double fn[14] = { 0, 1, 2 };

    do
    {
        printf("Input n (13>=n>=2): ");
        scanf("%d", &num);
    } while (num < 2 || num > 13);

    for (i = 3; i <= num; i++)
    {
        fn[i] = 2 * fn[i - 1] + fn[i - 2] * fn[i - 3];
    }
    printf("\nf(%d)=%.01f\n", num, fn[num]);

    return 0;
}

```

P732

DIFFICUTLY ★

输入 3 行 3 列的矩阵，输出所有元素的累加和。

可用素材: `printf("Please input the 3x3 Matrix:\n")`
`printf("\nsum=...\n"...`

程序的运行效果应类似地如图 732.1 所示，金色部分是从键盘输入的内容。



图 732.1

参考答案:

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

```
int main(void)
{
    int arr[3][3], i, j, sum = 0;

    printf("Please input the 3x3 Matrix:\n");
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }

    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++)
        {
            sum += arr[i][j];
        }
    }
    printf("\nsum=%d\n", sum);

    return 0;
}
```

P743

DIFFICUTLY ★

从键盘输入一行可带空格的字符串（约定：字符数 ≤ 127 字节），按逆序输出该字符串。

注：程序中不能使用库函数 `strrev()` 或使用同名的变量、函数、单词。

可用素材： `printf("Input a string: ")`
`printf("\nThe result is: ")`

程序的运行效果应类似地如图 743.1 所示，金色部分是从键盘输入的内容。



图 743.1

参考答案:

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

int main(void)
{
    char str[128];
    int len, i;

    printf("Input a string: ");
    gets(str);
    len = strlen(str);

    printf("\nThe result is: ");
    for (i = len - 1; i >= 0; i--)
    {
        printf("%c", str[i]);
    }
    putchar('\n');

    return 0;
}
```

P744

DIFFICULTLY ★

从键盘输入一个一百分制成绩，如果不在 0 ~ 100 范围内，则要求重新输入数据，直到输入的数据在 0 ~ 100 范围内。将输入的数据转换成等级“A”“B”“C”“D”“E”。

90 分以上为“A”，80 ~ 89 分为“B”，70 ~ 79 分为“C”，60 ~ 69 分为“D”，60 分以下为“E”，要求使用 switch()、case、default 语句且限制使用 5 个 case 分支，结果赋值给变量 *grade*，并将变量 *grade* 的值输出到屏幕上。

注：变量数据类型的选择应适当，在保证满足设计要求精度的情况下，养成不浪费内存空间和计算时间的好习惯。

可用素材： printf("please input the score(0-100): ")
printf("\nscore=...,grade=...

程序的运行效果应类似地如图 744.1 所示，金色部分是从键盘输入的内容。

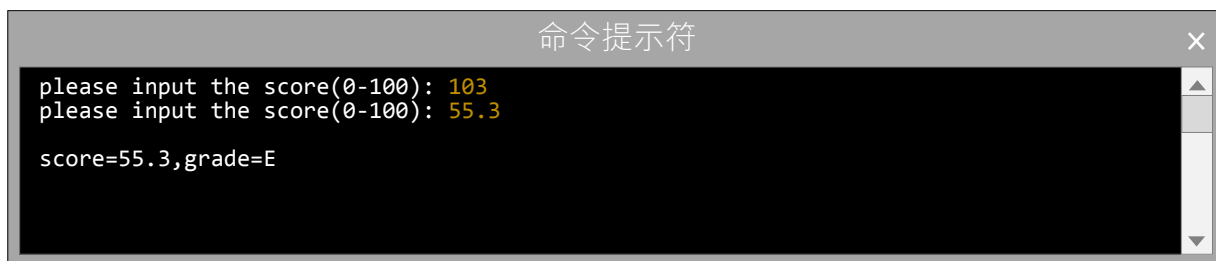


图 744.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    float score;
    int shi;
    char mark;
```

```

do
{
    printf("please input the score(0 - 100): ");
    scanf("%f", &score);
} while (score < 0 || score > 100);

shi = (int)score / 10;
switch (shi)
{
    case 10:
    case 9:
        mark = 'A';
        break;
    case 8:
        mark = 'B';
        break;
    case 7:
        mark = 'C';
        break;
    case 6:
        mark = 'D';
        break;
    default:
        mark = 'E';
}
printf("\nscore=%.1f,grade=%c\n", score, mark);

return 0;
}

```

P750

DIFFICUTLY ★

输入字符串 *s* (约定：字符数 ≤ 100 字节)，将字符串 *s* 中所有字符 “*” 删除，并将修改后的字符串显示出来。

可用素材： `printf("Please input a string: ")`
`printf("\nThe result is: ")`

程序的运行效果应类似地如图 750.1 所示，金色部分是从键盘输入的内容。



图 750.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    char str1[101], str2[101];
    int i, j;

    printf("Please input a string: ");
    gets(str1);

```

```

for (i = 0, j = 0; str1[i] != '\0'; i++)
{
    if (str1[i] != '*')
    {
        str2[j] = str1[i];
        j++;
    }
}
str2[j] = '\0';
printf("\nThe result is: %s\n", str2);

return 0;
}

```

P752

DIFFICUTLY ★

键盘输入 m 、 n (约定: m 和 n 均 ≤ 1000 且为正整数), 输出介于 m 和 n (含 m 和 n) 中能被 3 整除且至少有位数字是 5 的所有整数。

可用素材: `printf("Input m, n: ")`
`printf("\nResult: ")`

程序的运行效果应类似地如图 752.1 和图 752.2 所示, 金色部分是从键盘输入的内容。



图 752.1

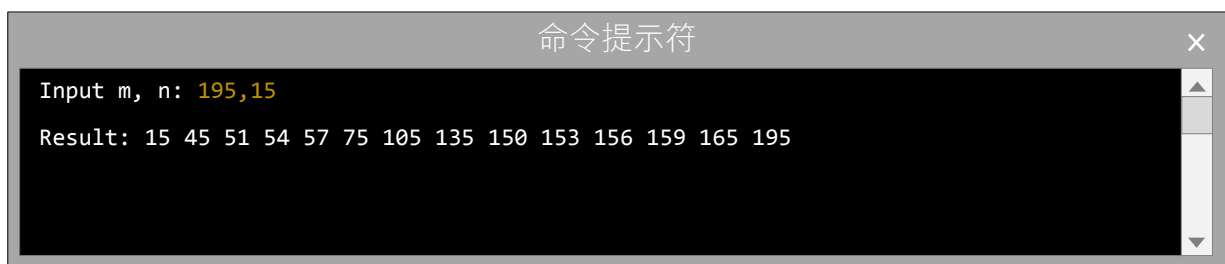


图 752.2

参考答案:

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

```

int main(void)
{
    int num1, num2, max, min, i, bai, shi, ge;

    do
    {
        printf("Input m, n: ");
        scanf("%d,%d", &num1, &num2);
    } while (num1 < 0 || num1 > 1000 || num2 < 0 || num2 > 1000);
    if (num1 > num2)
    {

```

```

        max = num1;
        min = num2;
    }
    else
    {
        max = num2;
        min = num1;
    }

    printf("\nResult: ");
    for (i = min; i <= max; i++)
    {
        if (i % 3 == 0)
        {
            if (i >= 10 && i < 100)
            {
                ge = i % 10;
                shi = i / 10;
                if (ge == 5 || shi == 5)
                {
                    printf("%d ", i);
                }
            }
            else if (i >= 100 && i < 1000)
            {
                ge = i % 10;
                shi = i / 10 % 10;
                bai = i / 100;
                if (ge == 5 || shi == 5 || bai == 5)
                {
                    printf("%d ", i);
                }
            }
        }
    }
    putchar('\n');

    return 0;
}

```

P753

DIFFICUTLY ★

计算 x 的 y 次方，其中 y 为整数（可以是负整数或 0）， x 为实型。

注：程序中不能使用库函数 `pow()` 或使用同名的变量、函数、单词。

可用素材： `printf("Input x, y: ")`
`printf("\nResult: ...^...=...")`

程序的运行效果应类似地如图 753.1 和图 753.2 所示，金色部分是从键盘输入的内容。



图 753.1



图 753.2

参考答案:

```
#include <stdio.h>

int main(void)
{
    double x0, result = 1;
    int y0, i;

    printf("Input x, y: ");
    scanf("%lf,%d", &x0, &y0);
    if (y0 > 0)
    {
        for (i = 1; i <= y0; i++)
        {
            result *= x0;
        }
    }
    else if (y0 < 0)
    {
        for (i = 1; i <= (-1) * y0; i++)
        {
            result *= x0;
        }
        result = 1 / result;
    }
    else
    {
        result = 1;
    }
    printf("\nResult: %f^%d=%f\n", x0, y0, result);

    return 0;
}
```

P800

DIFFICUTLY ★


程序 P800.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：求 $S = \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots + \frac{1}{N!}$ 并输出结果。 N 为任意自然数（只考虑 int 型），从键盘读入。

程序的运行效果应类似地如图 800.1 所示，金色部分是从键盘输入的内容。



图 800.1

附件:  P800.c

文件内容:

P800.c

```
#include <stdio.h>

void fun(double *sn, int n);

int main(void)
{
    int n;
    double Sum;

    printf("Please input n: ");
    scanf("%d", &n);

    <A> /* userCode(<30字符): 调用函数计算Sum */
    printf("\nS=1/1!+1/2!+...+1/%d!=%.16f\n", n, Sum);

    return 0;
}

void fun(double *sn, int n)
{
    int i;
    double Sum=0, jc=1;

    for (i=1; i<=n; i++)
    {
        jc *= i;
        Sum += 1 / jc;
    }

    <B> /* userCode(<30字符): 将计算结果通过指针参数返给主调函数 */
}
```

参考答案:

<A> fun(&Sum, n);
 *sn = Sum;

P830

DIFFICUTLY ★

求 $1 + 2 + 3 + \dots + n \leq m$ 时的最大 n 值及和 sum ($sum = 1 + 2 + 3 + \dots + n$)，其中 m 从键盘输入。
注：不得使用解方程、算平方根方法。

可用素材: printf("please input m: ")

```
printf("\nResult: n=..., sum=...
```

程序的运行效果应类似地如图 830.1 所示，金色部分是从键盘输入的内容。



图 830.1

参考答案:

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

```
int main(void)
```

```
{
```

```
    int num, sum = 0, i;
```

```
    printf("please input m: ");
```

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

```
    for (i = 1; ; i++)
```

```
    {
```

```
        sum += i;
```

```
        if (sum > num)
```

```
        {
```

```
            break;
```

```
        }
```

```
    }
```

```
    printf("\nResult: n=%d, sum=%d\n", i - 1, sum - i);
```

```
    return 0;
```

```
}
```

P215

DIFFICUTLY ★

求 $S = \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \dots + \frac{1}{N!}$ 并输出结果（显示时小数部分占 16 位，计算时要求从第 1 项开始往后累加）。 N 为任意自然数（只考虑 int 型），从键盘读入。

可用素材: `printf("Please input n: ")`

```
printf("\nS=1/1!+1/2!+...+1/...!=...
```

程序的运行效果应类似地如图 215.1 所示，金色部分是从键盘输入的内容。



图 215.1

参考答案:

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

```

int main(void)
{
    int num, i;
    double fac = 1, sum = 0;

    printf("Please input n: ");
    scanf("%d", &num);

    for (i = 1; i <= num; i++)
    {
        fac *= i;
        sum += 1 / fac;
    }
    printf("\nS=1/1!+1/2!+...+1/%d!=%.16f\n", num, sum);

    return 0;
}

```

P223

DIFFICUTLY ★

一个球从 10000 m 高度自由落下，每次落地后反弹回原高度的一半，再落下，再反弹。求它在第 n (n 从键盘输入) 次落地时，共经过多少米，第 n 次反弹多高。

可用素材： `printf("Please input n: ")`
`printf("\nsn=...,hn=...`

程序的运行效果应类似地如图 223.1 所示，金色部分是从键盘输入的内容。



图 223.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int num, i;
    double hn = 10000, sn = 10000;

    printf("Please input n: ");
    scanf("%d", &num);

    for (i = 1; i <= num; i++)
    {
        hn /= 2;
        sn += hn * 2;
    }
    sn -= hn * 2;
    printf("\nsn=%f,hn=%f\n", sn, hn);

    return 0;
}

```

P257

DIFFICUTLY ★


程序 P257.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘输入两个实数，分别保存到变量 *numA* 和 *numB*，调用函数 *swap()* 实现 *numA* 和 *numB* 的交换，并在 *main()* 函数中输出交换后的 *numA* 和 *numB*。

程序的运行效果应类似地如图 257.1 所示，金色部分是从键盘输入的内容。



图 257.1

附件:  P257.c

文件内容:

P257.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    float numA, numB;
```

```
    printf("please input numA, numB: ");
```

```
    scanf("%f,%f", &numA, &numB);
```

```
    <B>  /* userCode(<40字符): 调用函数实现numA和numB值的交换 */
```

```
    printf("\nnumA=%.3f, numB=%.3f\n", numA, numB);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<C> 
```

参考答案:

```
<A> void swap(float *num1, float *num2);
```

```
<B> swap(&numA, &numB);
```

```
<C> void swap(float *num1, float *num2)
```

```
{
```

```
    float tmp;
```

```
    tmp = *num1;
```

```
    *num1 = *num2;
```

```
    *num2 = tmp;
```

```
}
```

P321

DIFFICUTLY ★

在文本文件“Comp.txt”里有需要计算结果的整数算式，每个算式占一行且文件中有多个（数量不确定）算式，运算类型只有“加法（+）”或者“减法（-）”且运算符前后至少有一个空格。计算这些算式的结果并在屏幕上显示。

把程序运行时测试用的算式文件“Comp.txt”（见附件及文件内容）保存到程序 P321.c 所在的文件夹且文件名保持不变。

可用素材： `printf("Line %03d: %5d + %-5d = %-6d\n"...`
`printf("Line %03d: %5d - %-5d = %-6d\n"...`

程序的运行效果应类似地如图 321.1 所示。



图 321.1

附件： 
Comp.txt

文件内容：

Comp.txt

123 + 556
300 - 215
1001 - 18976
9123 + 5156

参考答案：

`#include <stdio.h>`

`int main(void)`
`{`

`FILE *fp;`

`int num1, num2, i = 1;`

`char op;`

`fp = fopen("Comp.txt", "r");`

`if (fp == NULL)`

`{`

`printf("File open error!\n");`

`return 0;`

`}`

`while (!feof(fp))`

`{`

`if (fscanf(fp, "%d %c %d", &num1, &op, &num2) == 3)`

`{`

`if (op == '+')`

`{`

`printf("Line %03d: %5d + %-5d = %-6d\n", i, num1, num2, num1 + num2);`

`}`

`if (op == '-')`

```

        {
            printf("Line %03d: %5d - %-5d = %-6d\n", i, num1, num2, num1 - num2);
        }
        i++;
    }

fclose(fp);

return 0;
}

```

P324

DIFFICUTLY ★

在文本文件“Comp.txt”“CompA.txt”“CompB.txt”里有需要计算结果的整数算式，文件“Comp.txt”提供参加运算的第一个数，文件“CompA.txt”提供进行运算的运算符（只有“加法(+)”或者“减法(-)”），文件“CompB.txt”提供参加运算的第二个数，每个数或运算符均占一行，组合起来成为一个算式，遇到无法组成一个完整算式时即结束运算。这样的算式有多个（数量不确定）。计算这些算式的结果并在屏幕上显示。




把程序运行时测试用的算式文件“Comp.txt”“CompA.txt”“CompB.txt”（见附件及文件内容）保存到程序 P324.c 所在的文件夹且文件名保持不变。

可用素材： `printf("Line %03d: %5d %c %-5d = %-6d\n"...`

程序的运行效果应类似地如图 324.1 所示。



图 324.1

附件：  Comp.txt  CompA.txt  CompB.txt

文件内容：

Comp.txt

123
300
1001
9123
9000

CompA.txt

+
-
-
+

CompB.txt

556
215

参考答案:

```
#include <stdio.h>

int main(void)
{
    FILE *fp1, *fp2, *fp3;
    int num1, num2, i = 1;
    char op;

    fp1 = fopen("Comp.txt", "r");
    fp2 = fopen("CompA.txt", "r");
    fp3 = fopen("CompB.txt", "r");
    if (fp1 == NULL || fp2 == NULL || fp3 == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    while (!feof(fp1) && !feof(fp2) && !feof(fp3))
    {
        if (fscanf(fp1, "%d", &num1) == 1 &&
            fscanf(fp2, "%c%c", &op) == 1 &&
            fscanf(fp3, "%d", &num2) == 1)
        {
            switch (op)
            {
                case '+':
                    printf("Line %03d: %5d %c %-5d = %-6d\n", i, num1, op, num2, num1 + num2);
                    break;
                case '-':
                    printf("Line %03d: %5d %c %-5d = %-6d\n", i, num1, op, num2, num1 - num2);
                    break;
                default:
                    printf("Error!\n");
            }
        }
        i++;
    }

    fclose(fp1);
    fclose(fp2);
    fclose(fp3);

    return 0;
}
```

P325

DIFFICUTLY ★

在文本文件“Comp.txt”“CompA.txt”“CompB.txt”里有需要计算结果的整数算式，文件“Comp.txt”提供参加运算的第一个数，文件“CompA.txt”提供进行运算的运算符（只有“加法(+)”或者“减法(-)”），文件“CompB.txt”提供参加运算的第二个数，每个数或运算符均占一行，组合起来成为一个算式，遇到无法组成一个完整算式时即结束运算。这样的算式有多个（数量不确定）。计算这些算式的结果并将结果以文本文件格式保存到程序 P325.c 所在的文件夹中且文件名命名为“CompC.txt”。

把程序运行时测试用的算式文件“Comp.txt”“CompA.txt”“CompB.txt”（见附件及文件内容）保存




到程序 P325.c 所在的文件夹且文件名保持不变。

可用素材： `fprintf(... "Line %03d: %5d %c %-5d = %-6d\n"...`

程序运行后生成的文件“CompC.txt”的内容应类似地如图 325.1 所示。



图 325.1 生成的文件“CompC.txt”之内容

附件：  Comp.txt  CompA.txt  CompB.txt

文件内容：

Comp.txt

123
300
1001
9123
9000

CompA.txt

+
-
-
+

CompB.txt

556
215
18976
5156

参考答案：

`#include <stdio.h>`

`int main(void)`

`{`

`FILE *fp1, *fp2, *fp3, *fp4;`

`int num1, num2, result, i = 1;`

`char op;`

`fp1 = fopen("Comp.txt", "r");`

`fp2 = fopen("CompA.txt", "r");`

`fp3 = fopen("CompB.txt", "r");`

`fp4 = fopen("CompC.txt", "w");`

`if (fp1 == NULL || fp2 == NULL || fp3 == NULL || fp4 == NULL)`

`{`

`printf("File open error!\n");`

`return 0;`

`}`

`while (!feof(fp1) && !feof(fp2) && !feof(fp3))`

`{`

`if (fscanf(fp1, "%d", &num1) == 1 &&`

```

        fscanf(fp2, "%C%c", &op) == 1 &&
        fscanf(fp3, "%d", &num2) == 1)
    {
        switch (op)
        {
            case '+':
                result = num1 + num2;
                fprintf(fp4, "Line %03d: %5d %c %-5d = %-6d\n", i, num1, op, num2, result);
                break;
            case '-':
                result = num1 - num2;
                fprintf(fp4, "Line %03d: %5d %c %-5d = %-6d\n", i, num1, op, num2, result);
                break;
            default:
                fprintf(fp4, "Error!\n");
        }
    }
    i++;
}

fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);

return 0;
}

```

P327

DIFFICUTLY ★ | NO REMARKS

程序 P327.c 已编写部分代码（见文件内容，标准版和增强版），请根据程序中的要求完善程序，实现指定的功能要求。

附件:  P327-Standard.c  P327-Plus.c

文件内容:

P327-Standard.c

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

```
<A> ??pointerExample(??)
```

```
int main(void)
```

```

{
    int tstNum = 500;
    char tstCh='A', tstStr[]="Hello, C programmer.....";
    <B>
    // 以上变量定义可增加。

    // 以下仅写一条语句，调用函数 pointerExample，实现：
    // 修改 tstNum 的值为 501，修改 tstCh 的值为 'B'，
    // 修改字符串 tstStr 的内容为 "I am fine, thank you very very much!"。
    // 通过函数值得到 pointerExample 中的字符串 staStr，
    // 获取 pointerExample 中的 autoNum 的值。
    <C>

    // 完善以下的 printf（一条语句），输出 tstNum 和 tstCh 的值、串 tstStr、
    // pointerExample 中 autoNum 的值及串 staStr。
    <D> _printf(??)
}

```

```

        return 0;
    }

    <E> ??pointerExample(??)
    {
        static char *staStr = "Hello, pointer!\n";
        int autoNum = 2001;

        <F>
    }

```

P327-Plus.c

```

#include <stdio.h>

    <A> ??pointerExample(??)

int main(void)
{
    int tstNum = 500;
    char tstCh='A', tstStr[]="Hello, C programmer.....";
    <B>
    // 以上变量定义可增加。

    // 以下仅写一条语句，调用函数 pointerExample，实现：
    // 修改 tstNum 的值为 501，修改 tstCh 的值为 'B'，
    // 修改字符串 tstStr 的内容为 "I am fine, thank you very very much!"。
    // 通过函数值得到 pointerExample 中的字符串 staStr，
    // 获取 pointerExample 中的 autoNum 的值和 staNum1 的地址。
    <C>

    // 以下仅写一条语句，实现：将 pointerExample 中的 staNum1 的值改为 3002。
    <D>

    // 完善以下的 printf（一条语句），输出 tstNum 和 tstCh 的值、串 tstStr、
    // pointerExample 中 autoNum 和 staNum1 的值及串 staStr。
    <E> printf(??)

    return 0;
}

    <F> ??pointerExample(??)
    {
        static int staNum1 = 3001;
        static char *staStr = "Hello, pointer!\n";
        int autoNum = 2001;

        <G>
    }

```

参考答案：

P327-Standard.c

```

<A> void pointerExample(int *ptstNum, char *ptstCh, char tstStr[], char tmp[], int *pfunNum,
    char funStr[]);
<B> char tmp[] = "I am fine, thank you very very much!"; funStr[100];
    int funNum;
<C> pointerExample(&tstNum, &tstCh, tstStr, tmp, &funNum, funStr);
<D> printf("%d\n%c\n%s\n%d\n%s\n", tstNum, tstCh, tstStr, funNum, funStr);

```

```

<E> void pointerExample(int *ptstNum, char *ptstCh, char tstStr[], char tmp[], int *pfunNum,
char funStr[])
<F> *ptstNum = 501;
    *ptstCh = 'B';

    while (*tstStr != '\0')
    {
        *tstStr = *tmp;
        tstStr++;
        tmp++;
    }

    *pfunNum = autoNum;

    while (*staStr != '\0')
    {
        *funStr = *staStr;
        funStr++;
        staStr++;
    }
    *funStr = '\0';

```

P327-Plus.c

```

<A> int *pointerExample(int *ptstNum, char *ptstCh, char tstStr[], char tmp[], int *pfunNum,
char funStr[]);
<B> char tmp[] = "I am fine, thank you very very much!"; funStr[100];
    int funNum, *pstaNum1;
<C> pstaNum1 = pointerExample(&tstNum, &tstCh, tstStr, tmp, &funNum, funStr);
<D> *pstaNum1 = 3002;
<E> printf("%d\n%c\n%s\n%d\n%d\n%s\n", tstNum, tstCh, tstStr, funNum, *pstaNum1, funStr);
<F> int *pointerExample(int *ptstNum, char *ptstCh, char tstStr[], char tmp[], int *pfunNum,
char funStr[])
<G> *ptstNum = 501;
    *ptstCh = 'B';

    while (*tstStr != '\0')
    {
        *tstStr = *tmp;
        tstStr++;
        tmp++;
    }

    *pfunNum = autoNum;

    while (*staStr != '\0')
    {
        *funStr = *staStr;
        funStr++;
        staStr++;
    }
    *funStr = '\0';

    return &staNum1;

```

P328

DIFFICUTLY ★

程序运行时，先从键盘输入一个文本文件的文件名（约定：字符数 ≤ 127 字节，可含路径），再在屏幕上显示该文件的内容。

把程序运行时测试用的文件“Test.txt”（见附件及文件内容）保存到计算机。

可用素材: `printf("input the file's name: ")`
`printf("\nfile open error!")`
`printf("-----File Begin:-----\n")`
`printf("\n----- File End. -----\n")`

程序的运行效果应类似地如图 328.1 所示，金色部分是从键盘输入的内容。

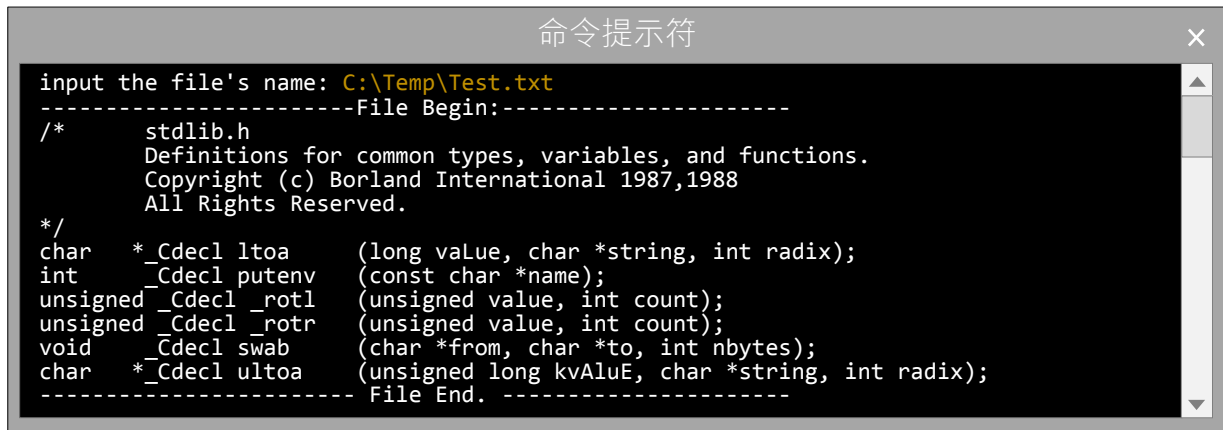


图 328.1

附件:  Test.txt

文件内容:

Test.txt

```
/*      stdlib.h
Definitions for common types, variables, and functions.
Copyright (c) Borland International 1987,1988
All Rights Reserved.
*/
char    *_Cdecl ltoa      (long vaLue, char *string, int radix);
int      _Cdecl putenv    (const char *name);
unsigned _Cdecl _rotl     (unsigned value, int count);
unsigned _Cdecl _rotr     (unsigned value, int count);
void     _Cdecl swab      (char *from, char *to, int nbytes);
char     *_Cdecl ultoa    (unsigned long kvAluE, char *string, int radix);
----- File End. -----
```

参考答案:

`#include <stdio.h>`

```
int main(void)
{
```

```
    FILE *fp;
    char str[128];
    int ch;
```

```
    printf("input the file's name: ");
    gets(str);
```

```
    fp = fopen(str, "r");
    if (fp == NULL)
    {
        printf("file open error!\n");
        return 0;
    }
```

```
    printf("-----File Begin:-----\n");
    while ((ch = fgetc(fp)) != EOF)
```

```

    {
        putchar(ch);
    }
    printf("\n----- File End. ----- \n");

    fclose(fp);

    return 0;
}

```

P330

DIFFICUTLY ★

程序运行时，先从键盘输入一个文本文件的文件名（约定：字符数 ≤ 127 字节，可含路径），再在屏幕上显示该文件的内容。注意，对于文件中的字符“*”，在屏幕上改为显示字符“@”。

把程序运行时测试用的文件“Test.txt”（见附件及文件内容）保存到计算机。

可用素材：

```

printf("input the file's name: ")
printf("\nfile open error!")
printf("-----File Begin:----- \n")
printf("\n----- File End. ----- \n")

```

程序的运行效果应类似地如图 330.1 所示，金色部分是从键盘输入的内容。



图 330.1

附件：

Test.txt

文件内容：

Test.txt

```

/*      stdlib.h
Definitions for common types, variables, and functions.
Copyright (c) Borland International 1987,1988
All Rights Reserved.

*/
char    *_Cdecl ltoa      (long value, char *string, int radix);
int      _Cdecl putenv    (const char *name);
unsigned _Cdecl _rotl     (unsigned value, int count);
unsigned _Cdecl _rotr     (unsigned value, int count);
void     _Cdecl swab      (char *from, char *to, int nbytes);
char     *_Cdecl ultoa    (unsigned long value, char *string, int radix);

```

参考答案：

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

```

int main(void)
{
    FILE *fp;
    char str[128];
    int ch;

    printf("input the file's name: ");
    gets(str);

    fp = fopen(str, "r");
    if (fp == NULL)
    {
        printf("file open error!\n");
        return 0;
    }

    printf("-----File Begin:-----\n");
    while ((ch = fgetc(fp)) != EOF)
    {
        if (ch == '*')
        {
            putchar('@');
        }
        else
        {
            putchar(ch);
        }
    }
    printf("\n----- File End. ----- \n");

    fclose(fp);

    return 0;
}

```

P716

DIFFICUTLY ★

求 $s = a + aa + aaa + aaaa + aa...a$ 的值，其中 a 是一个数字（可取 1~9 之间的一个值）。例如 $2 + 22 + 222 + 2222 + 22222$ （此时共有 5 个数相加），其中 a 值和有几个数相加由键盘输入控制。注意 s 的值有可能超出 int 的范围。

可用素材： `printf("Please input a,n: ")`
`printf("\na+aa+...=...")`

程序的运行效果应类似地如图 716.1 所示，金色部分是从键盘输入的内容。



图 716.1

参考答案：

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

```
int main(void)
```

```

{
    int an, num, i;
    double sum = 0, bn = 0;

    printf("Please input a,n: ");
    scanf("%d,%d", &an, &num);

    for (i = 1; i <= num; i++)
    {
        bn += an;
        sum += bn;
        bn *= 10;
    }
    printf("\na+aa+...=%.0f\n", sum);

    return 0;
}

```

P745

DIFFICUTLY ★

输入两个正整数 m 和 n ，求其最大公约数和最小公倍数。

注：最大公约数也称最大公因子，指某几个整数共有因子中最大的一个；两个整数公有的倍数称为它们的公倍数，其中最小的一个正整数称为它们两个的最小公倍数。

可用素材： `printf("please input two integer numbers: ")`
`printf("\nthe greatest common divisor is ...`
`printf("\nthe least common multiple is ...`

程序的运行效果应类似地如图 745.1 所示，金色部分是从键盘输入的内容。

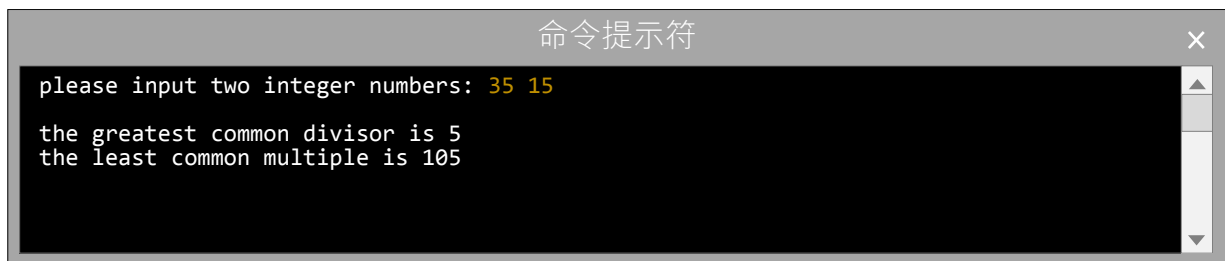


图 745.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int num1, num2, big, small, yue, bei, i;

    printf("please input two integer numbers: ");
    scanf("%d%d", &num1, &num2);
    if (num1 > num2)
    {
        big = num1;
        small = num2;
    }
    else
    {
        big = num2;
        small = num1;
    }
}

```



```

for (i = small; i >= 1; i--)
{
    if (big % i == 0 && small % i == 0)
    {
        yue = i;
        break;
    }
}
bei = big * small / yue;
printf("\nthe greatest common divisor is %d", yue);
printf("\nthe least common multiple is %d\n", bei);

return 0;
}

```

P747

DIFFICUTLY ★

先从键盘上读入 15 个数放在一个数组 $a[15]$ 中，然后再输入一个数，要求找出该数是数组中第几个（从 0 开始计数，若有相同的数，则计首次出现的数）元素 $a[i]$ 的值。如果该数不在数组中，则输出相应的提示信息。

可用素材：

```

printf("please input 15 integer numbers:\n")
printf("please input the integer you want to find: ")
printf("\n%d has been found,it is a[%d]\n"...
printf("\n%d has not been found\n"...

```

程序的运行效果应类似地如图 747.1 和图 747.2 所示，金色部分是从键盘输入的内容。



图 747.1



图 747.2

参考答案：

```

#include <stdio.h>

int main(void)
{
    int arr[15], num, i, flag = 0;

    printf("please input 15 integer numbers:\n");
    for (i = 0; i < 15; i++)

```

```

{
    scanf("%d", &arr[i]);
}

printf("please input the integer you want to find: ");
scanf("%d", &num);

for (i = 0; i < 15; i++)
{
    if (arr[i] == num)
    {
        printf("\n%d has been found,it is a[%d]\n", num, i);
        flag = 1;
        break;
    }
}
if (flag == 0)
{
    printf("\n%d has not been found\n", num);
}

return 0;
}

```

P803

DIFFICUTLY ★


程序 P803.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘输入两个整数，分别保存到变量 *numA* 和 *numB*，调用函数 *swap()* 实现 *numA* 和 *numB* 的交换，并在 *main()* 函数中输出交换过后的 *numA* 和 *numB*。

程序的运行效果应类似地如图 803.1 所示，金色部分是从键盘输入的内容。



图 803.1

附件:  P803.c

文件内容:

P803.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    int numA, numB;
```

```
    printf("please input numA, numB: ");
```

```
    scanf("%d,%d", &numA, &numB);
```

```

    <B>          /* userCode(<40字符): 调用函数实现numA和numB值的交换 */
    printf("\nnnumA=%d, numB=%d\n", numA, numB);

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>

```

参考答案:

```

<A> void swap(int *num1, int *num2);
<B> swap(&numA, &numB);
<C> void swap(int *num1, int *num2)
    {
        int tmp;

        tmp = *num1;
        *num1 = *num2;
        *num2 = tmp;
    }

```

P806

DIFFICUTLY ★

程序 P806.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：键盘输入 m , n （约定： m 和 n 均 ≤ 1000 且为正整数），输出介于 m 和 n （含 m 和 n ）中能被 3 整除且至少有位数字是 5 的所有整数。程序中函数 void fun(int rsNum[], int begin, int end, int *Count) 的功能是：计算出介于 $begin$ 和 end （含 $begin$ 和 end ）、能被 3 整除且至少有位数字是 5 的所有整数，并按从小到大的顺序放在 **rsNum** 所指的数组中，这些数的个数通过形参 *Count* 返回。

程序的运行效果应类似地如图 806.1 和图 806.2 所示，金色部分是从键盘输入的内容。



图 806.1

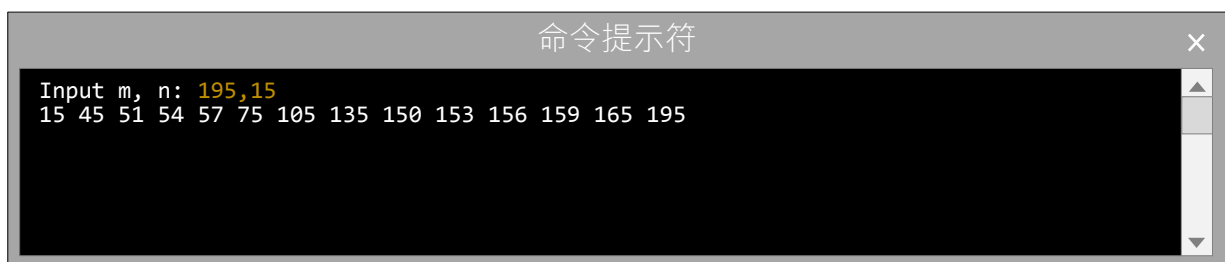


图 806.2

附件:  P806.c

文件内容:

P806.c

```
#include<stdio.h>

void fun(int rsNum[], int begin, int end, int *Count);

int main(void)
{
    int i, m, n, tmp, num[1000], numCount;

    printf("Input m, n: ");
    scanf("%d,%d", &m, &n);
    if (m > n)
    {
        tmp = m;
        m = n;
        n = tmp;
    }

    /* 本部分代码功能建议: 调用函数fun()完成计算 */
    /* User Code Begin(Limit: lines<=1, lineLen<=50, 考生可在本行后添加代码、最多1行、行长<=50字符) */
    <A>
    /* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

    for (i=0; i<numCount; i++)
    {
        printf("%d ", num[i]);
    }
    putchar('\n');

    return 0;
}

void fun(int rsNum[], int begin, int end, int *Count)
{
    int i, j=0, *numP=rsNum;

    for (i=begin; i<=end; i++)
    {
        if (i/100 == 5 || i/10 % 10 == 5 || i%10 == 5)
        {
            numP[j] = i;
            j++;
        }
    }

    /* User Code Begin(考生可在本行后添加代码, 行数不限) */
    <B>
    /* User Code End(考生添加代码结束) */
}
/* Program End(程序到此结束, 此后不能添加内容, 否则0分) */
```

参考答案:

```
<A> fun(num, m, n, &numCount);
<B> for (i = 0, *Count = 0; i < j; i++)
{
    if (numP[i] % 3 == 0)
```

```

    {
        rsNum[*Count] = numP[i];
        *Count += 1;
    }
}

```

P816

DIFFICUTLY ★

程序 P816.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：输入两个整数 m 和 n ，输出大于等于 m ($m > 5$) 的 n 个素数，输出的各素数间以空格相隔。

注：素数（Prime Number），亦称质数，指在一个大于 1 的自然数中，除了 1 和此整数自身外，没法被其他自然数整除的数。

程序的运行效果应类似地如图 816.1 所示，金色部分是从键盘输入的内容。



图 816.1

附件:  P816.c

文件内容:

P816.c

```

#include <stdio.h>
#include <math.h>

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int m, n, cnt;

    printf("Input the m, n: ");
    scanf("%d,%d", &m, &n);

    printf("\nThe result:\n");
    for (cnt=0; cnt<n; m++)
    {
        <B> /* userCode(<50字符): 调用函数判断m是否为素数 */
        {
            printf("%d ", m);
            cnt++;
        }
    }
    putchar('\n');

    return 0;
}

```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
<C>
```

参考答案:

```
<A> int Check(int num);
<B> if (Check(m) == 0)
<C> int Check(int num)
{
    int i, flag = 0;

    for (i = 2; i < num; i++)
    {
        if (num % i == 0)
        {
            flag = 1;
            break;
        }
    }

    if (flag == 0)
    {
        return 0;
    }
    else
    {
        return EOF;
    }
}
```

P831

DIFFICUTLY ★

某班有 40 位同学参加考试，成绩（整数）从键盘输入，求全班最高分、最低分以及平均分，并统计该班同学的考试及格率。

可用素材： `printf("请输入40位同学的成绩: ")`
`printf("\n最高分: ...\n最低分: ...\n平均分: ...\n及格率: ...`

程序的运行效果应类似地如图 831.1 所示，金色部分是从键盘输入的内容。

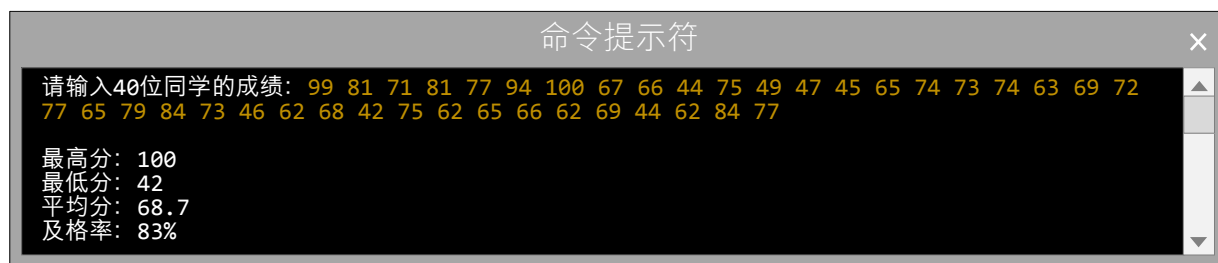


图 831.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int score[40], i, max = 0, min = 100, sum = 0, pass = 0;
    float average, ratio;
```

```

printf("请输入40位同学的成绩: ");
for (i = 0; i < 40; i++)
{
    scanf("%d", &score[i]);
}

for (i = 0; i < 40; i++)
{
    if (score[i] > max)
    {
        max = score[i];
    }
    if (score[i] < min)
    {
        min = score[i];
    }
    if (score[i] >= 60)
    {
        pass++;
    }
    sum += score[i];
}
average = (float)sum / 40;
ratio = (float)pass / 40 * 100;
printf("\n最高分: %d\n最低分: %d\n平均分: %.1f\n及格率: %.0f%%\n", max, min, average, ratio);

return 0;
}

```

P239

DIFFICUTLY ★

先从键盘读入若干个整数（读到 -1 或读满 16 个数均结束读入），然后倒序输出这些数。

可用素材： `printf("请输入若干个数: ")`
`printf("\n这些数倒序为: ")`

程序的运行效果应类似地如图 239.1 和图 239.2 所示，金色部分是从键盘输入的内容。

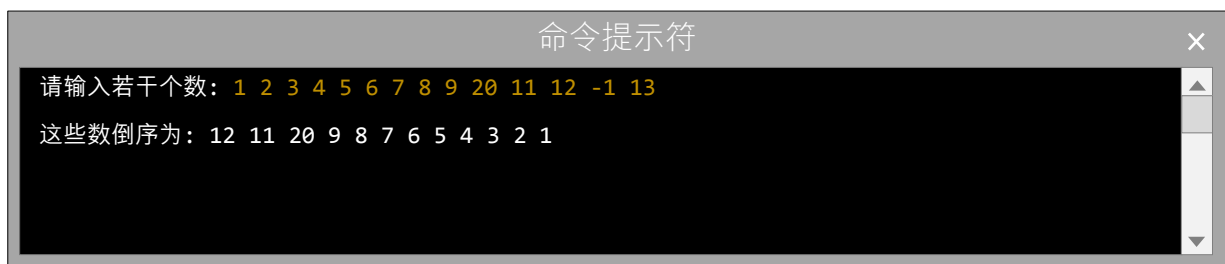


图 239.1 读到 -1 时的情况

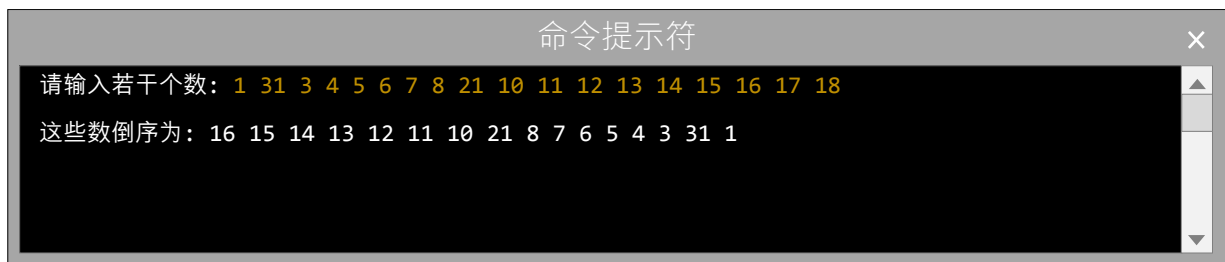


图 239.2 读满 16 个数时的情况

参考答案:

```
#include <stdio.h>

int main(void)
{
    int arr[16], i, tmp, count = 0;

    printf("请输入若干个数: ");
    for (i = 0; i < 16; i++)
    {
        scanf("%d", &tmp);
        if (tmp != -1)
        {
            arr[count] = tmp;
            count++;
        }
        else
        {
            break;
        }
    }

    printf("\n这些数倒序为: ");
    for (i = count - 1; i >= 0; i--)
    {
        printf("%d ", arr[i]);
    }
    putchar('\n');

    return 0;
}
```

P241

DIFFICUTLY ★

程序 P241.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数输出数组 **arrA** 和 **arrB** 的各元素。

程序的运行效果应类似地如图 241.1 所示，金色部分是从键盘输入的内容。



图 241.1

附件:



P241.c

文件内容:

P241.c

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

```

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int arrA[5], arrB[8];

    printf("请输入5个数: ");
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
    printf("请输入8个数: ");
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],
    &arrB[6], &arrB[7]);

    printf("\narrA = ");
    <B> /* userCode(<30字符): 调用函数输出arrA的所有元素 */
    printf("\narrB = ");
    <C> /* userCode(<30字符): 调用函数输出arrB的所有元素 */

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<D>

```

参考答案:

```

<A> void Print(int arr[], int n);
<B> Print(arrA, 5);
<C> Print(arrB, 8);
<D> void Print(int arr[], int n)
{
    int i;

    for (i = 0; i < n; i++)
    {
        printf("%d ", arr[i]);
    }
}

```

P243

DIFFICULTLY ★

程序 P243.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数计算数组 **arrA** 和 **arrB** 各元素的平均值、再输出平均值。

程序的运行效果应类似地如图 243.1 所示，金色部分是从键盘输入的内容。



图 243.1

附件:  P243.c

文件内容:

P243.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> _____
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8];
```

```
    float avgA, avgB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    <B> _____ /* userCode(<30字符): 调用函数计算arrA所有元素的平均值 */
```

```
    printf("\nAvg(arrA) = %.1f", avgA);
```

```
    <C> _____ /* userCode(<30字符): 调用函数计算arrB所有元素的平均值 */
```

```
    printf("\nAvg(arrB) = %.1f\n", avgB);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<D> _____
```

参考答案:

```
<A> float GetAvg(int arr[], int n);
```

```
<B> avgA = GetAvg(arrA, 5);
```

```
<C> avgB = GetAvg(arrB, 8);
```

```
<D> float GetAvg(int arr[], int n)
```

```
{
```

```
    int i, sum = 0;
```

```
    float avg;
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        sum = sum + arr[i];
```

```
    }
```

```
    avg = (float)sum / n;
```

```
    return avg;
```

```
}
```

P244

DIFFICUTLY ★

程序 P244.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个欲查找的数 **searchVal**，然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找 **searchVal** 所在位置的下标（不考虑在数组中存在多个 **searchVal** 的情况）、输出查找结果。

程序的运行效果应类似地如图 244.1 所示，金色部分是从键盘输入的内容。



图 244.1

附件:  P244.c

文件内容:

P244.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], searchVal, positionA, positionB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    printf("请输入一个欲查找的数: ");
```

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

```
<B> /* userCode(<50字符): 调用函数查找searchVal在arrA中的下标 */
```

```
if (-1 == positionA)
```

```
{
```

```
    printf("\narrA: not Find!");
```

```
}
```

```
else
```

```
{
```

```
    printf("\narrA: position = %d", positionA);
```

```
}
```

```
<C> /* userCode(<50字符): 调用函数查找searchVal在arrB中的下标 */
```

```
if (-1 == positionB)
```

```
{
```

```
    printf("\narrB: not Find!\n");
```

```
}
```

```
else
```

```
{
```

```

        printf("\narrB: position = %d\n", positionB);
    }

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
<D>

```

参考答案:

```

<A> int Search(int arr[], int n, int searchVal);
<B> positionA = Search(arrA, 5, searchVal);
<C> positionB = Search(arrB, 8, searchVal);
<D> int Search(int arr[], int n, int searchVal)
    {
        int i;


        for (i = 0; i < n; i++)
        {
            if (arr[i] == searchVal)
            {
                return i;
            }
        }

        return -1;
    }

```

P245

DIFFICUTLY ★

程序 P245.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数计算数组 **arrA** 和 **arrB** 各元素的最大值、再输出最大值。

程序的运行效果应类似地如图 245.1 所示，金色部分是从键盘输入的内容。

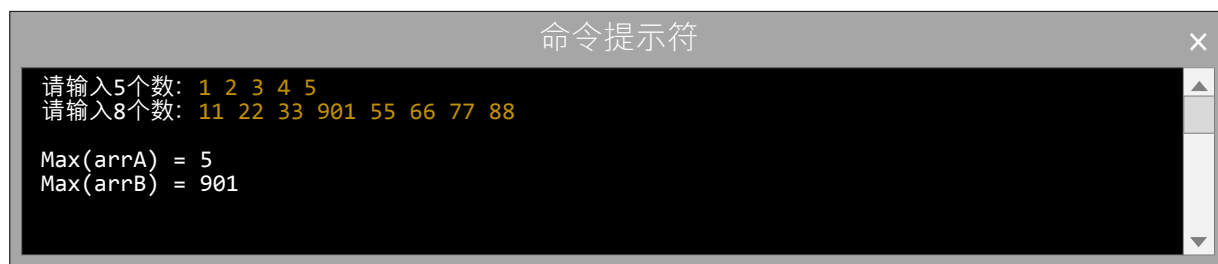


图 245.1

附件:  P245.c

文件内容:

P245.c

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

```
/* userCode(<50字符>): 自定义函数之原型声明 */
```

```
<A> 
```

```

int main(void)
{
    int arrA[5], arrB[8], maxA, maxB;

    printf("请输入5个数: ");
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
    printf("请输入8个数: ");
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],
    &arrB[6], &arrB[7]);

    <B>          /* userCode(<30字符): 调用函数计算arrA中元素的最大值 */
    printf("\nMax(arrA) = %d", maxA);
    <C>          /* userCode(<30字符): 调用函数计算arrB中元素的最大值 */
    printf("\nMax(arrB) = %d\n", maxB);

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<D>

```

参考答案:

```

<A> int GetMax(int arr[], int n);
<B> maxA = GetMax(arrA, 5);
<C> maxB = GetMax(arrB, 8);
<D> int GetMax(int arr[], int n)
{
    int max = arr[0], i;


    for (i = 1; i < n; i++)
    {
        if (arr[i] > max)
        {
            max = arr[i];
        }
    }

    return max;
}

```

P246

DIFFICUTLY ★


程序 P246.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数计算数组 **arrA** 和 **arrB** 各元素的最小值、再输出最小值。

程序的运行效果应类似地如图 246.1 所示，**金色部分**是从键盘输入的内容。



图 246.1

附件:  P246.c

文件内容:

P246.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> _____
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], minA, minB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    <B> _____ /* userCode(<30字符): 调用函数计算arrA中元素的最小值 */
```

```
    printf("\nMin(arrA) = %d", minA);
```

```
    <C> _____ /* userCode(<30字符): 调用函数计算arrB中元素的最小值 */
```

```
    printf("\nMin(arrB) = %d\n", minB);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<D> _____
```

参考答案:

```
<A> int GetMin(int arr[], int n);
```

```
<B> minA = GetMin(arrA, 5);
```

```
<C> minB = GetMin(arrB, 8);
```

```
<D> int GetMin(int arr[], int n)
```

```
{
```

```
    int min = arr[0], i;
```

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

```
    {
```

```
        if (arr[i] < min)
```

```
        {
```

```
            min = arr[i];
```

```
        }
```

```
    }
```

```
    return min;
```

```
}
```

P247

DIFFICUTLY ★

程序 P247.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数计算数组 **arrA** 和 **arrB** 各元素最大值（不考虑有多个最大值的情况）所在位置的下标、再输出该下标。

程序的运行效果应类似地如图 247.1 所示，金色部分是从键盘输入的内容。

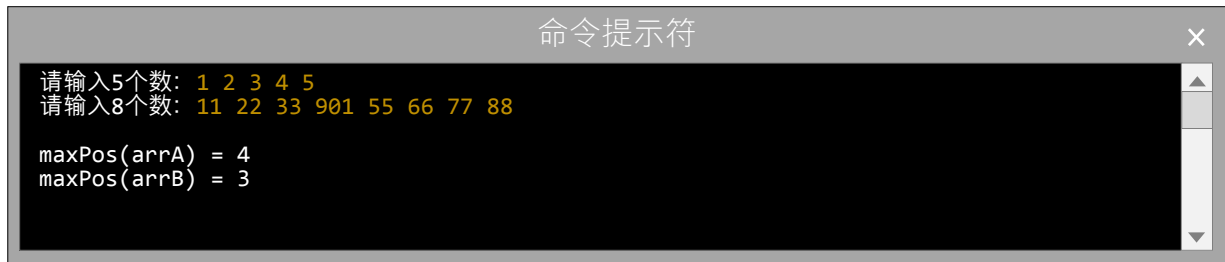



图 247.1

附件:  P247.c

文件内容:

P247.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], maxPosA, maxPosB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    <B>  /* userCode(<30字符): 调用函数计算arrA中元素的最大值所在的下标 */
```

```
    printf("\nmaxPos(arrA) = %d", maxPosA);
```

```
    <C>  /* userCode(<30字符): 调用函数计算arrA中元素的最大值所在的下标 */
```

```
    printf("\nmaxPos(arrB) = %d\n", maxPosB);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<D> 
```

参考答案:

```
<A> int GetMaxPos(int arr[], int n);
```

```
<B> maxPosA = GetMaxPos(arrA, 5);
```

```
<C> maxPosB = GetMaxPos(arrB, 8);
```

```
<D> int GetMaxPos(int arr[], int n)
```

```
{
```

```

    int max = arr[0], maxi = 0, i;

    for (i = 1; i < n; i++)
    {
        if (arr[i] > max)
        {
            max = arr[i];
            maxi = i;
        }
    }

    return maxi;
}

```

P248

DIFFICUTLY ★


程序 P248.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数计算数组 **arrA** 和 **arrB** 各元素最小值（不考虑有多个最小值的情况）所在位置的下标、再输出该下标。

程序的运行效果应类似地如图 248.1 所示，金色部分是从键盘输入的内容。



图 248.1

附件:  P248.c

文件内容:

P248.c

```

#include <stdio.h>

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int arrA[5], arrB[8], minPosA, minPosB;

    printf("请输入5个数: ");
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
    printf("请输入8个数: ");
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],
    &arrB[6], &arrB[7]);

    <B> /* userCode(<30字符): 调用函数计算arrA中元素的最小值所在的下标 */
    printf("\nminPos(arrA) = %d", minPosA);
    <C> /* userCode(<30字符): 调用函数计算arrA中元素的最小值所在的下标 */
    printf("\nminPos(arrB) = %d\n", minPosB);

    return 0;
}

```



```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<D>
```

参考答案:

```
<A> int GetMinPos(int arr[], int n);
<B> minPosA = GetMinPos(arrA, 5);
<C> minPosB = GetMinPos(arrB, 8);
<D> int GetMinPos(int arr[], int n)
{
    int min = arr[0], mini = 0, i;

    for (i = 1; i < n; i++)
    {
        if (arr[i] < min)
        {
            min = arr[i];
            mini = i;
        }
    }

    return mini;
}
```

P249

DIFFICUTLY ★

程序 P249.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个数 *num*，然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找大于 *num* 的元素个数、输出查找结果。

程序的运行效果应类似地如图 249.1 所示，金色部分是从键盘输入的内容。



图 249.1

附件:  P249.c

文件内容:

P249.c

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

```
/* userCode(<70字符>): 自定义函数之原型声明 */
```

```
<A> _____
```

```
int main(void)
```

```
{
```

```

int arrA[5], arrB[8], num, countA, countB;

printf("请输入5个数: ");
scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
printf("请输入8个数: ");
scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],
&arrB[6], &arrB[7]);
printf("请再输入一个数: ");
scanf("%d", &num);

<B>          /* userCode(<50字符): 调用函数查找arrA中大于num的元素个数 */
printf("\nCount(arrA) = %d", countA);
<C>          /* userCode(<50字符): 调用函数查找arrB中大于num的元素个数 */
printf("\nCount(arrB) = %d\n", countB);

return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<D>

```

参考答案:

```

<A> int GetCount(int arr[], int n, int num);
<B> countA = GetCount(arrA, 5, num);
<C> countB = GetCount(arrB, 8, num);
<D> int GetCount(int arr[], int n, int num)
{
    int i, count = 0;


    for (i = 0; i < n; i++)
    {
        if (arr[i] > num)
        {
            count++;
        }
    }

    return count;
}

```

P250

DIFFICUTLY ★


程序 P250.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个数 *num*，然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找小于 *num* 的元素个数、输出查找结果。

程序的运行效果应类似地如图 250.1 所示，金色部分是从键盘输入的内容。



图 250.1

附件:  P250.c

文件内容:

P250.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], num, countA, countB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    printf("请再输入一个数: ");
```

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

```
    <B> /* userCode(<50字符): 调用函数查找arrA中小于num的元素个数 */
```

```
    printf("\nCount(arrA) = %d", countA);
```

```
    <C> /* userCode(<50字符): 调用函数查找arrB中小于num的元素个数 */
```

```
    printf("\nCount(arrB) = %d\n", countB);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<D>
```

参考答案:

```
<A> int GetCount(int arr[], int n, int num);
```

```
<B> countA = GetCount(arrA, 5, num);
```

```
<C> countB = GetCount(arrB, 8, num);
```

```
<D> int GetCount(int arr[], int n, int num)
```

```
{
```

```
    int i, count = 0;
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        if (arr[i] < num)
```

```
        {
```

```
            count++;
```

```
        }
```

```
    }
```

```
        return count;
    }
```

P251

DIFFICUTLY ★

程序 P251.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找大于其平均值的元素个数、输出查找结果。

程序的运行效果应类似地如图 251.1 所示，金色部分是从键盘输入的内容。



图 251.1

附件:  P251.c

文件内容:

P251.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], countA, countB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    <B> /* userCode(<50字符): 调用函数查找arrA中大于其平均值的元素个数 */
```

```
    printf("\nCount(arrA) = %d", countA);
```

```
    <C> /* userCode(<50字符): 调用函数查找arrB中大于其平均值的元素个数 */
```

```
    printf("\nCount(arrB) = %d\n", countB);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<D>
```

参考答案:

```
<A> int GetCount(int arr[], int n);
```

```
<B> countA = GetCount(arrA, 5);
```

```

<C> countB = GetCount(arrB, 8);
<D> int GetCount(int arr[], int n)
{
    int i, sum = 0, count = 0;
    float avg;

    for (i = 0; i < n; i++)
    {
        sum += arr[i];
    }
    avg = (float)sum / n;

    for (i = 0; i < n; i++)
    {
        if ((float)arr[i] > avg)
        {
            count++;
        }
    }

    return count;
}

```

P252

DIFFICUTLY ★


程序 P252.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找小于其平均值的元素个数、输出查找结果。

程序的运行效果应类似地如图 252.1 所示，金色部分是从键盘输入的内容。



图 252.1

附件:  P252.c

文件内容:

P252.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], countA, countB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```

    scanf("%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],
&arrB[6], &arrB[7]);

    <B>          /* userCode(<50字符): 调用函数查找arrA中小于其平均值的元素个数 */
    printf("\nCount(arrA) = %d", countA);
    <C>          /* userCode(<50字符): 调用函数查找arrB中小于其平均值的元素个数 */
    printf("\nCount(arrB) = %d\n", countB);

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<D>

```

参考答案:

```

<A> int GetCount(int arr[], int n);
<B> countA = GetCount(arrA, 5);
<C> countB = GetCount(arrB, 8);
<D> int GetCount(int arr[], int n)
{
    int i, sum = 0, count = 0;
    float avg;

    for (i = 0; i < n; i++)
    {
        sum += arr[i];
    }
    avg = (float)sum / n;


    for (i = 0; i < n; i++)
    {
        if ((float)arr[i] < avg)
        {
            count++;
        }
    }

    return count;
}

```

P253

DIFFICUTLY ★

程序 P253.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个数 *num*（这些数均 ≥ 0 ），然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找大于 *num* 的最小数、输出查找结果。

程序的运行效果应类似地如图 253.1 所示，金色部分是从键盘输入的内容。



图 253.1

附件:  P253.c

文件内容:

P253.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], num, minA, minB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    printf("请再输入一个数: ");
```

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

```
<B>
```

```
/* userCode(<50字符): 调用函数查找在arrA中大于num的最小数 */
```

```
if (-1 == minA)
```

```
{
```

```
    printf("\narrA: not Find!");
```

```
}
```

```
else
```

```
{
```

```
    printf("\nMin(arrA) = %d", minA);
```

```
}
```

```
<C>
```

```
/* userCode(<50字符): 调用函数查找在arrB中大于num的最小数 */
```

```
if (-1 == minB)
```

```
{
```

```
    printf("\narrB: not Find!\n");
```

```
}
```

```
else
```

```
{
```

```
    printf("\nMin(arrB) = %d\n", minB);
```

```
}
```

```
return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<D>
```

参考答案:


```
<A> int GetMin(int arr[], int n, int num);
<B> minA = GetMin(arrA, 5, num);
<C> minB = GetMin(arrB, 8, num);
<D> int GetMin(int arr[], int n, int num)
{
    int result = -1, i, flag = 0;

    for (i = 0; i < n; i++)
    {
        if (flag == 0 && arr[i] > num)
        {
            result = arr[i];
            flag = 1;
        }
        if (flag == 1 && arr[i] > num && arr[i] < result)
        {
            result = arr[i];
        }
    }

    return result;
}
```

P254

DIFFICUTLY ★


程序 P254.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个数 **num**，然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找大于 **num** 的元素个数、输出查找结果。

程序的运行效果应类似地如图 254.1 所示，金色部分是从键盘输入的内容。



图 254.1

附件:  P254.c

文件内容:

P254.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], num, maxA, maxB;
```

```
    printf("请输入5个数: ");
```

```

scanf("%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
printf("请输入8个数: ");
scanf("%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],
&arrB[6], &arrB[7]);
printf("请再输入一个数: ");
scanf("%d", &num);

<B> /* userCode(<50字符): 调用函数查找在arrA中小于num的最大数 */
if (-1 == maxA)
{
    printf("\narrA: not Find!");
}
else
{
    printf("\nMax(arrA) = %d", maxA);
}

<C> /* userCode(<50字符): 调用函数查找在arrB中小于num的最大数 */
if (-1 == maxB)
{
    printf("\narrB: not Find!\n");
}
else
{
    printf("\nMax(arrB) = %d\n", maxB);
}

return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<D>

```

参考答案:

```

<A> int GetMax(int arr[], int n, int num);
<B> maxA = GetMax(arrA, 5, num);
<C> maxB = GetMax(arrB, 8, num);
<D> int GetMax(int arr[], int n, int num)
{
    int result = -1, i, flag = 0;

    for (i = 0; i < n; i++)
    {
        if (flag == 0 && arr[i] < num)
        {
            result = arr[i];
            flag = 1;
        }
        if (flag == 1 && arr[i] < num && arr[i] > result)
        {
            result = arr[i];
        }
    }

    return result;
}

```

P255

DIFFICUTLY ★


程序 P255.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个数 **num**（这些数均 ≥ 0 ），然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找小于 **num** 的最大数（不考虑有多个最大值的情况）所在位置的下标、再输出该下标。

程序的运行效果应类似地如图 255.1 所示，金色部分是从键盘输入的内容。



图 255.1

附件:  P255.c

文件内容:

P255.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], num, minPosA, minPosB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5], &arrB[6], &arrB[7]);
```

```
    printf("请再输入一个数: ");
```

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

```
<B>  /* userCode(<50字符): 调用函数查找在arrA中大于num的最小数所在的下标 */
```

```
    if (-1 == minPosA)
```

```
    {
```

```
        printf("\narrA: not Find!");
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("\nminPos(arrA) = %d", minPosA);
```

```
    }
```

```
<C>  /* userCode(<50字符): 调用函数查找在arrB中大于num的最小数所在的下标 */
```

```
    if (-1 == minPosB)
```

```
    {
```

```
        printf("\narrB: not Find!\n");
```

```
    }
```

```
    else
```

```
    {
```

```

        printf("\nminPos(arrB) = %d\n", minPosB);
    }

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
<D>

```

参考答案:

```

<A> int GetMinPos(int arr[], int n, int num);
<B> minPosA = GetMinPos(arrA, 5, num);
<C> minPosB = GetMinPos(arrB, 8, num);
<D> int GetMinPos(int arr[], int n, int num)
    {
        int resultpos = -1, result = -1, i, flag = 0;

        for (i = 0; i < n; i++)
        {
            if (flag == 0 && arr[i] > num)
            {
                result = arr[i];
                resultpos = i;
                flag = 1;
            }
            if (flag == 1 && arr[i] > num && arr[i] < result)
            {
                result = arr[i];
                resultpos = i;
            }
        }

        return resultpos;
    }
}

```

P256

DIFFICUTLY ★


程序 P256.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：从键盘分别读入 5 个数到 **arrA** 中、8 个数至 **arrB** 中，再读入一个数 *num*（这些数均 ≥ 0 ），然后分别调用自定义函数在数组 **arrA** 和 **arrB** 中查找小于 *num* 的最大数（不考虑有多个最大值的情况）所在位置的下标、再输出该下标。

程序的运行效果应类似地如图 256.1 所示，金色部分是从键盘输入的内容。



图 256.1

附件: 
P256.c

文件内容:

P256.c

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

```
/* userCode(<70字符): 自定义函数之原型声明 */
```

```
<A> _____
```

```
int main(void)
```

```
{
```

```
    int arrA[5], arrB[8], num, maxPosA, maxPosB;
```

```
    printf("请输入5个数: ");
```

```
    scanf("%d%d%d%d%d", &arrA[0], &arrA[1], &arrA[2], &arrA[3], &arrA[4]);
```

```
    printf("请输入8个数: ");
```

```
    scanf("%d%d%d%d%d%d%d", &arrB[0], &arrB[1], &arrB[2], &arrB[3], &arrB[4], &arrB[5],  
    &arrB[6], &arrB[7]);
```

```
    printf("请再输入一个数: ");
```

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

```
    <B> _____ /* userCode(<50字符): 调用函数查找在arrA中小于num的最大数所在的下标 */
```

```
    if (-1 == maxPosA)
```

```
    {
```

```
        printf("\narrA: not Find!");
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("\nmaxPos(arrA) = %d", maxPosA);
```

```
    }
```

```
    <C> _____ /* userCode(<50字符): 调用函数查找在arrB中小于num的最大数所在的下标 */
```

```
    if (-1 == maxPosB)
```

```
    {
```

```
        printf("\narrB: not Find!\n");
```

```
    }
```

```
    else
```

```
    {
```

```
        printf("\nmaxPos(arrB) = %d\n", maxPosB);
```

```
    }
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<D> _____
```

参考答案:

```
<A> int GetMaxPos(int arr[], int n, int num);
```

```
<B> maxPosA = GetMaxPos(arrA, 5, num);
```

```
<C> maxPosB = GetMaxPos(arrB, 8, num);
```

```
<D> int GetMaxPos(int arr[], int n, int num)
```

```
{
```

```
    int resultpos = -1, result = -1, i, flag = 0;
```

```
    for (i = 0; i < n; i++)
```

```
    {
```

```
        if (flag == 0 && arr[i] < num)
```

```
        {
```

```

        result = arr[i];
        resultpos = i;
        flag = 1;
    }
    if (flag == 1 && arr[i] < num && arr[i] > result)
    {
        result = arr[i];
        resultpos = i;
    }
}

return resultpos;
}

```

P792

DIFFICULTLY ★

将命令行输入的三个字符串按从小到大的顺序排序后输出。

注意事项:

1. 命令行格式为: P792 **str₁** **str₂** **str₃**, 当命令行格式不正确 (参数个数不为 4) 时, 应报错。
2. 程序的返回值 (即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值, 也称退出代码) 规定为:
 - a) 正常运行结束时, 返回 0。
 - b) 命令行格式不对, 返回 1。

可用素材: printf(" error, usage: P792 str1 str2 str3\n")
 printf(" output: %s %s %s\n"...

程序的运行效果应类似地如图 792.1 所示, 橙色部分为系统命令行提示符, 表示程序 P792.exe 所在的文件夹, 用户的程序位置可不必如此; 金色部分是从命令行输入的内容。



图 792.1

参考答案:

```

#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    char *tmp;
    int i, j;

    if (argc != 4)
    {
        printf(" error, usage: P792 str1 str2 str3\n");
        return 1;
    }
}

```

```

    }

    for (j = 1; j < 3; j++)
    {
        for (i = 1; i < 4 - j; i++)
        {
            if (strcmp(argv[i], argv[i + 1]) > 0)
            {
                tmp = argv[i];
                argv[i] = argv[i + 1];
                argv[i + 1] = tmp;
            }
        }
    }

    printf("    output: %s    %s    %s\n", argv[1], argv[2], argv[3]);
    return 0;
}

```

P811

DIFFICUTLY ★

程序 P811.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：输入三角形的三边长 a 、 b 、 c ，求三角形面积 $area$ ，并输出。如果输入的三边构不成三角形，应给出“data error”的信息提示。

注：根据“海伦—秦九韶”公式， $area = \sqrt{p(p-a)(p-b)(p-c)}$ ，其中 $p = \frac{a+b+c}{2}$ 。

程序的运行效果应类似地如图 811.1 和图 811.2 所示，金色部分是从键盘输入的内容。

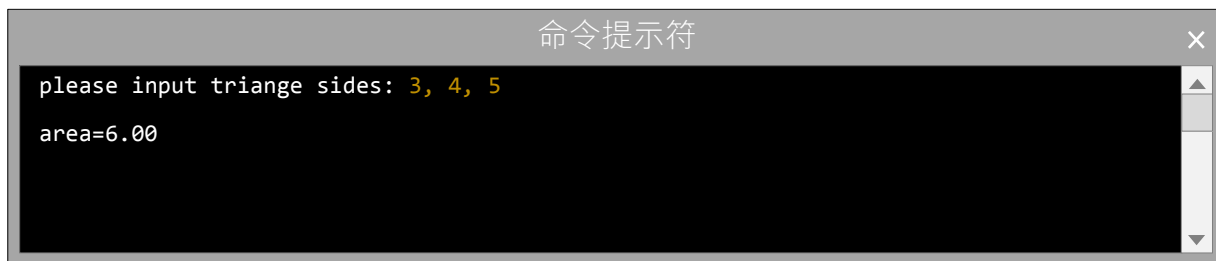


图 811.1



图 811.2

附件:  P811.c

文件内容:

P811.c

```

#include <math.h>
#include <stdio.h>

```

```

/* userCode(<80字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    double bianA, bianB, bianC, mianJi;

    printf("please input triange sides: ");
    scanf("%lf,%lf,%lf", &bianA, &bianB, &bianC);

    if (bianA<0 || bianB<0 || bianC<0
        || (bianA+bianB <= bianC) || (bianA+bianC <= bianB) || (bianB+bianC <= bianA))
    {
        printf("\ndata error\n");
    }
    else
    {
        <B> /* userCode(<50字符): 调用函数计算三角形面积 */
        printf("\narea=%.2f\n", mianJi);
    }

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>

```

参考答案:

```

<A> double GetArea(double a0, double b0, double c0);
<B> mianJi = GetArea(bianA, bianB, bianC);
<C> double GetArea(double a0, double b0, double c0)
{
    double p0, sum;

    p0 = (a0 + b0 + c0) / 2;
    sum = sqrt(p0 * (p0 - a0) * (p0 - b0) * (p0 - c0));

    return sum;
}

```

P812

DIFFICUTLY ★

程序 P812.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：输入三角形的三边长 a 、 b 、 c （约定为整数），求三角形面积 $area$ ，并输出。如果输入的三边构不成三角形，应给出“data error”的信息提示。

注：根据“海伦—秦九韶”公式， $area = \sqrt{p(p-a)(p-b)(p-c)}$ ，其中 $p = \frac{a+b+c}{2}$ 。

程序的运行效果应类似地如图 812.1 和图 812.2 所示，金色部分是从键盘输入的内容。

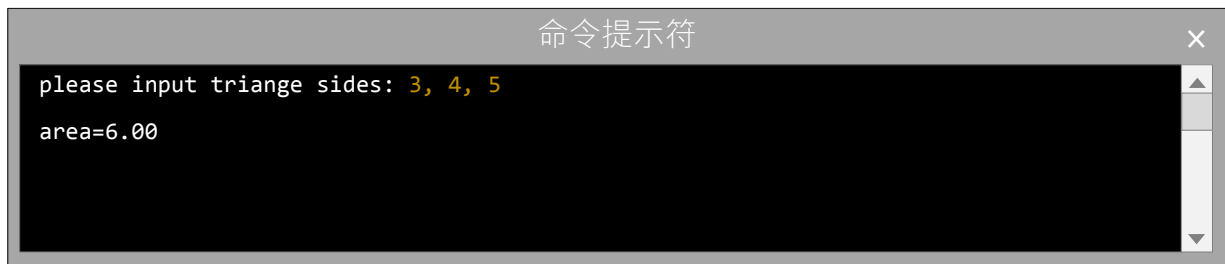


图 812.1

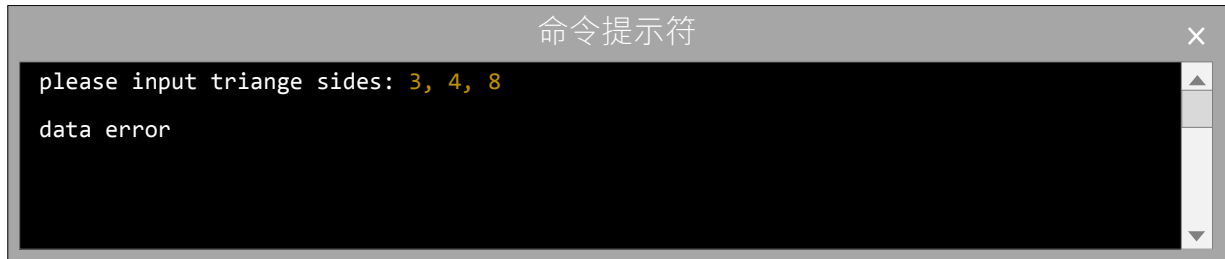


图 812.2

附件:  P812.c

文件内容:

P812.c

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

/* userCode(<80字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int aBian, bBian, cBian;
    double mJi;

    printf("please input triange sides: ");
    scanf("%d,%d,%d", &aBian, &bBian, &cBian);

    if (aBian<0 || bBian<0 || cBian<0
        || (aBian+bBian <= cBian) || (aBian+cBian <= bBian) || (bBian+cBian <= aBian))
    {
        printf("\ndata error\n");
    }
    else
    {
        <B> /* userCode(<50字符): 调用函数计算三角形面积 */
        printf("\narea=%.2f\n", mJi);
    }

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>
```

参考答案:

```
<A> double GetArea(int a0, int b0, int c0);
<B> mJi = GetArea(aBian, bBian, cBian);
```



```
<C> double GetArea(int a0, int b0, int c0)
{
    double p0, sum;

    p0 = (double)(a0 + b0 + c0) / 2;
    sum = sqrt(p0 * (p0 - a0) * (p0 - b0) * (p0 - c0));

    return sum;
}
```

P813

DIFFICUTLY ★


程序 P813.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：有一递推数列，满足 $f(0) = 0$, $f(1) = 1$, $f(2) = 2$, $f(n+1) = 2f(n) + f(n-1)f(n-2)$ ($n \geq 2$)，编写程序求 $f(n)$ 的值（ n 由键盘输入， $13 \geq n \geq 2$ ）。

程序的运行效果应类似地如图 813.1 所示，金色部分是从键盘输入的内容。



图 813.1

附件:  P813.c

文件内容:

P813.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> _____
```

```
int main(void)
```

```
{
```

```
    int n;
```

```
    double fn;
```

```
    printf("Input n (13>=n>=2): ");
```

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

```
    <B> _____ /* userCode(<50字符): 调用函数计算fn */
```

```
    printf("\nf(%d)=%.0f\n", n, fn);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<C> _____
```

参考答案:

```
<A> double Sum(int num);
<B> fn = Sum(n);
<C> double Sum(int num)
{
    double sum;

    if (num == 0)
    {
        sum = 0;
    }
    else if (num == 1)
    {
        sum = 1;
    }
    else if (num == 2)
    {
        sum = 2;
    }
    else
    {
        sum = 2 * Sum(num - 1) + Sum(num - 2) * Sum(num - 3);
    }

    return sum;
}
```

P814

DIFFICUTLY ★

程序 P814.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）

程序的功能是：输入 3 行 3 列的矩阵，输出所有元素的累加和。

程序的运行效果应类似地如图 814.1 所示，金色部分是从键盘输入的内容。



图 814.1

附件:  P814.c

文件内容:

P814.c

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

```
/* userCode(<50字符>): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
    int num[3][3], i, j, sum;
```

```

printf("Please input the 3x3 Matrix:\n");
for (i=0; i<3; i++)
{
    for (j=0; j<3; j++)
    {
        scanf("%d", &num[i][j]);
    }
}

<B> /* userCode(<50字符): 调用函数计算矩阵所有元素之和 */
printf("\nsum=%d\n", sum);
return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>

```

参考答案:

```

<A> int Sum(int num[3][3]);
<B> sum = Sum(num);
<C> int Sum(int num[3][3])
{
    int i, j, sum = 0;

    for (j = 0; j < 3; j++)
    {
        for (i = 0; i < 3; i++)
        {
            sum += num[i][j];
        }
    }

    return sum;
}

```

P817

DIFFICUTLY ★

程序 P817.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：从键盘输入一行可带空格的字符串（约定：字符数 ≤ 127 字节），按逆序输出该字符串。

注：程序中不能使用库函数 `strrev()` 或使用同名的变量、函数、单词。

程序的运行效果应类似地如图 817.1 所示，金色部分是从键盘输入的内容。



图 817.1

附件:  P817.c

文件内容:

P817.c

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

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    char str[128];

    printf("Input a string: ");
    gets(str);

    <B> /* userCode(<50字符): 调用函数将字符串str逆序存放 */
    printf("\nThe result is: %s\n", str);
    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>
```

参考答案:

```
<A> void Invert(char str[]);
<B> Invert(str);
<C> void Invert(char str[])
{
    int len, i, j, tmp;

    len = strlen(str);

    for (i = 0, j = len - 1; i < j; i++, j--)
    {
        tmp = str[i];
        str[i] = str[j];
        str[j] = tmp;
    }
}
```

P820

DIFFICUTLY ★

程序 P820.c 已编写部分代码 (见文件内容), 请根据程序中的要求完善程序 (在指定的位置添加代码或将 换成代码)。

程序的功能是: 从键盘输入一个整数 n , 计算对应的函数 $f(n)$ 值, 并按示例格式输出相应信息。函数 $f(n)$ 的定义如下:

$$f(n) = \begin{cases} 1 & (n = 1) \\ 2f(n/2) + n & (n > 1) \\ 0 & (n < 1) \end{cases}$$

程序的运行效果应类似地如图 820.1 和图 820.2 所示，金色部分是从键盘输入的内容。

命令提示符

Please input a number: -3
f(-3) = 0

图 820.1

命令提示符

Please input a number: 6
f(6) = 16

图 820.2

附件:  P820.c

文件内容:

P820.c

```
#include <stdio.h>

/* userCode(<50字符): 自定义函数之原型声明 */
<A>_____

int main(void)
{
    int n;
    long int Fn;

    printf("Please input a number: ");
    scanf("%d", &n);

    <B>_____ /* userCode(<50字符): 调用函数计算f(n) */
    printf("\nf(%d) = %ld\n", n, Fn);
    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>_____
```

参考答案:

```
<A> long int GetFn(int n);
<B> Fn = GetFn(n);
<C> long int GetFn(int n)
{
    long int Fn;

    if (n == 1)
    {
        Fn = 1;
    }
}
```

```

        else if (n > 1)
        {
            Fn = 2 * GetFn(n / 2) + n;
        }
        else
        {
            Fn = 0;
        }

    return Fn;
}

```

P821

DIFFICUTLY ★

程序 P821.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。


程序的功能是：从键盘输入一个整数 n ($n \geq 0$) 和 x ，计算对应的 n 阶勒让德多项式 $P_n(x)$ 的值，并按示例格式输出相应信息。 n 阶勒让德多项式 $P_n(x)$ 的定义如下：

$$P_n(x) = \begin{cases} 1 & (n = 0) \\ x & (n = 1) \\ \frac{(2n-1)xP_{n-1}(x) - (n-1)P_{n-2}(x)}{n} & (n > 1) \end{cases}$$

程序的运行效果应类似地如图 821.1 所示，金色部分是从键盘输入的内容。



图 821.1

附件:  P821.c

文件内容:

P821.c

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

```
/* userCode(<50字符>): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    double Pnx;
```

```
    int n, x;
```

```
    printf("please input n, x: ");
```

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

```
    <B>
```

```
    /* userCode(<50字符>): 调用函数计算Pn(x) */
```

```
    printf("\nThe answer is %.6f.\n", Pnx);
```

```

        return 0;
    }

    /* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
    <C>

```

参考答案:

```

<A> double GetPnx(int n, int x);
<B> Pnx = GetPnx(n, x);
<C> double GetPnx(int n, int x)
{
    double Pnx;

    if (n == 0)
    {
        Pnx = 1;
    }
    else if (n == 1)
    {
        Pnx = x;
    }
    else if (n > 1)
    {
        Pnx = ((2 * n - 1) * x * GetPnx(n - 1, x) - (n - 1) * GetPnx(n - 2, x)) / n;
    }

    return Pnx;
}

```

P822

DIFFICUTLY ★

程序 P822.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：计算并输出 n ($n \leq 100$) 门课程的平均成绩。

程序的运行效果应类似地如图 822.1 所示，金色部分是从键盘输入的内容。

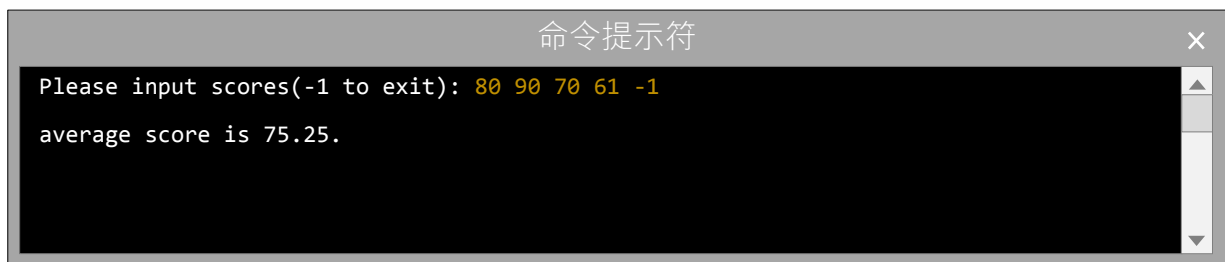


图 822.1

附件:  P822.c

文件内容:

```

P822.c
#include <stdio.h>

#define maxNums 100

```

```

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int i, count, scores[maxNums];
    float avgScore;

    printf("Please input scores(-1 to exit): ");
    for (i=0; i<maxNums; i++)
    {
        scanf("%d", &scores[i]);
        if (-1 == scores[i])
        {
            break;
        }
    }
    count = i;

    <B> /* userCode(<50字符): 调用函数计算平均成绩 */
    printf("\naverage score is %.2f.\n", avgScore);
    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>

```

参考答案:

```

<A> float GetAvg(int scores[], int n);
<B> avgScore = GetAvg(scores, i);
<C> float GetAvg(int scores[], int n)
{
    int sum = 0, i;
    float avg;

    for (i = 0; i < n; i++)
    {
        sum += scores[i];
    }
    avg = (float)sum / i;

    return avg;
}

```

P824

DIFFICUTLY ★

程序 P824.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：求 $S = \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots + \frac{1}{N!}$ 并输出结果。 N 为任意自然数（只考虑 int 型），从键盘读入。

程序的运行效果应类似地如图 824.1 所示，金色部分是从键盘输入的内容。



图 824.1

附件:  P824.c

文件内容:

P824.c

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

```
/* 本部分代码功能建议: 函数原型声明 */
```

```
/* User Code Begin(Limit: lines<=1, lineLen<=20, 考生可在本行后添加代码、最多1行、行长<=20字符) */
```

```
<A>
```

```
/* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */
```

```
int main(void)
```

```
{
```

```
    int i, n;
```

```
    double Sum = 0;
```

```
    printf("Please input n: ");
```

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

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

```
    {
```

```
        Sum += 1 / fac();
```

```
    }
```

```
    printf("\nS=1/1!+1/2!+...+1/%d!=%.16f\n", n, Sum);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin(考生在此后根据设计需要完成程序的其它部分, 如函数的定义, 行数不限) */
```

```
<B>
```

参考答案:

```
<A> double fac(void);
```

```
<B> double fac(void)
```

```
{
```

```
    static int i = 1;
```

```
    static double result = 1;
```

```
    result *= i;
```

```
    i++;
```

```
    return result;
```

```
}
```

P240

DIFFICUTLY ★

程序 P240.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：分 3 次调用自定义函数实现从键盘为数组 **arrayA**、**arrayB**、**arrayC** 分别读入 6、10、15 个数并计算每一组数的和，然后分别输出每一数组头尾两个元素的值及所有元素的和。

可用素材： `printf("请输入%d个数: "...`

程序的运行效果应类似地如图 240.1 所示，金色部分是从键盘输入的内容。

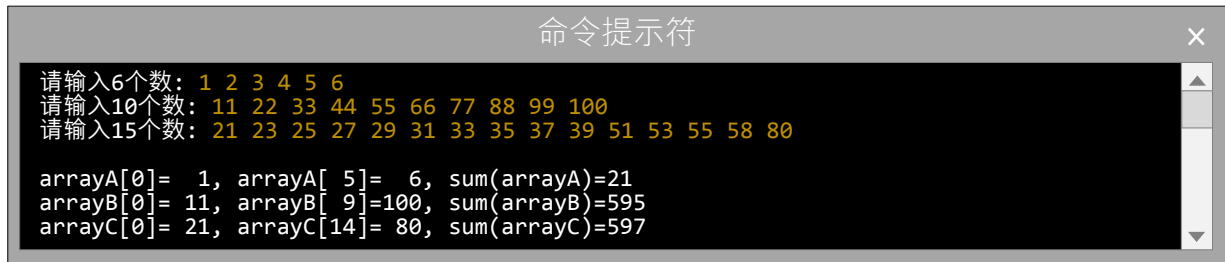



图 240.1

附件:  P240.c

文件内容:

P240.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A> 
```

```
int main(void)
```

```
{
```

```
    int arrayA[6], arrayB[10], arrayC[15], sumA, sumB, sumC;
```

```
    <B>  /* userCode(<50字符): 调用函数读 6个数到arrayA中, 并计算和 */
```

```
    <C>  /* userCode(<50字符): 调用函数读10个数到arrayB中, 并计算和 */
```

```
    <D>  /* userCode(<50字符): 调用函数读15个数到arrayC中, 并计算和 */
```

```
    printf("\narrayA[0]=%3d, arrayA[ 5]=%3d, sum(arrayA)=%d", arrayA[0], arrayA[5], sumA);
```

```
    printf("\narrayB[0]=%3d, arrayB[ 9]=%3d, sum(arrayB)=%d", arrayB[0], arrayB[9], sumB);
```

```
    printf("\narrayC[0]=%3d, arrayC[14]=%3d, sum(arrayC)=%d\n", arrayC[0], arrayC[14], sumC);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<E> 
```

参考答案:

```
<A> int Sum(int arr[], int n);
```

```
<B> sumA = Sum(arrayA, 6);
```

```
<C> sumB = Sum(arrayB, 10);
```

```
<D> sumC = Sum(arrayC, 15);
```

```
<E> int Sum(int arr[], int n)
```

```
{
```

```
    int sum = 0, i;
```

```

printf("请输入%d个数: ", n);
for (i = 0; i < n; i++)
{
    scanf("%d", &arr[i]);
    sum += arr[i];
}

return sum;
}

```

P242

DIFFICUTLY ★

程序 P242.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：从键盘读入若干个整数（读到 -1 或读满 16 个数均结束读入），然后倒序输出这些数。程序的运行效果应类似地如图 242.1 和图 242.2 所示，金色部分是从键盘输入的内容。

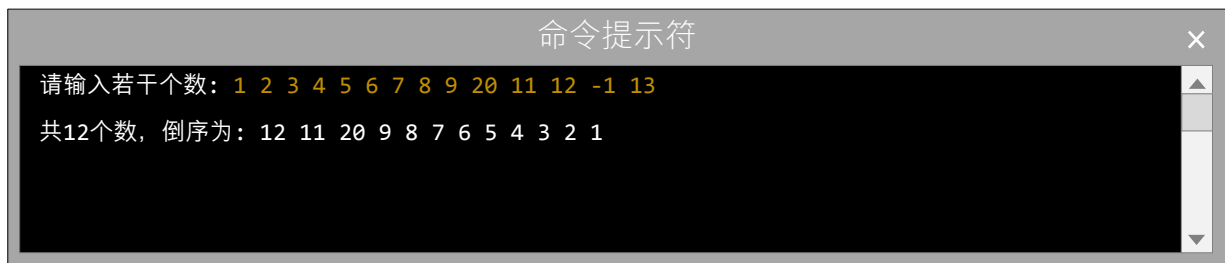


图 242.1 读到 -1 时的情况

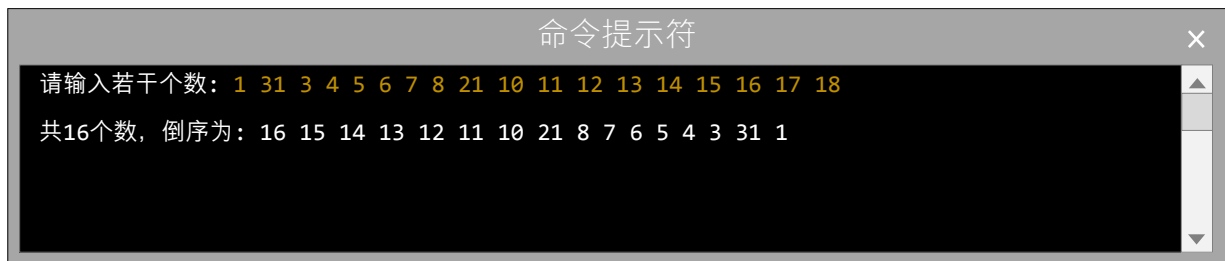



图 242.2 读满 16 个数时的情况

附件:  P242.c

文件内容:

P242.c

```

#include <stdio.h>

<A> _____ /* userCode(<50字符): 自定义函数1之原型声明 */
<B> _____ /* userCode(<50字符): 自定义函数2之原型声明 */

int main(void)
{
    int num[16], count;

    printf("请输入若干个整数: ");
    <C> _____ /* userCode(<30字符): 调用函数读入数据到num中并统计个数 */

    printf("\n共%d个数，倒序为: ", count);
    <D> _____ /* userCode(<30字符): 调用函数倒序输出num的所有元素 */
}

```

```
        return 0;
    }

/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
<E>
```

参考答案:

```
<A> int GetValue(int num[]);
<B> void Invert(int num[], int count);
<C> count = GetValue(num);
<D> Invert(num, count);
<E> int GetValue(int num[])
{
    int i, count = 0;

    for (i = 0; i < 16; i++)
    {
        scanf("%d", &num[i]);
        if (num[i] == -1)
        {
            break;
        }
        count++;
    }


    return count;
}

void Invert(int num[], int count)
{
    int i;

    for (i = count - 1; i >= 0; i--)
    {
        printf("%d ", num[i]);
    }
}
```

P789

DIFFICUTLY ★

程序 P789.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘上读入一行字符，删除除英文字母“A~Z”“a~z”外的所有其它字符，并输出剩余的字符。

注：要求用指针完成函数中数组参数的传递、以及各个数组元素的访问，且函数中不得再定义和使用数组，即自定义函数头和函数体中不得出现数组下标形式的表示法。

程序的运行效果应类似地如图 789.1 所示，金色部分是从键盘输入的内容。



图 789.1

附件:  P789.c

文件内容:

P789.c

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

```
/* userCode(<80字符>): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    char str[100];
```

```
    printf("Please input the string : ");
```

```
    gets(str);
```

```
    deleteother(str);
```

```
    printf("\noutput: %s\n", str);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<B>
```

参考答案:

```
<A> void deleteother(char str[]);
```

```
<B> void deleteother(char str[])
```

```
{
```

```
    char *p1, *p2;
```

```
    p1 = str;
```

```
    p2 = str;
```

```
    while (*p2 != '\0')
```

```
    {
```

```
        if (*p2 >= 'A' && *p2 <= 'Z' || *p2 >= 'a' && *p2 <= 'z')
```

```
        {
```

```
            *p1 = *p2;
```

```
            p1++;
```

```
        }
```

```
        p2++;
```

```
    }
```

```
    *p1 = '\0';
```

```
    str = p1;
```

```
}
```

P793

DIFFICUTLY ★

从键盘读入一个数 n ，先逆序输出 n 的各位数，再输出 n 的各位数之和。

可用素材： `printf("请输入一个数: ")`
`printf("\n该数的各位数之逆序为: ")`
`printf("\n该数的各位数之和为: %d\n"...`

程序的运行效果应类似地如图 793.1 所示，金色部分是从键盘输入的内容。



图 793.1

参考答案 1 (常规)：

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

int main(void)
{
    int num, len, sum = 0, i;
    char str[10];

    printf("请输入一个数: ");
    scanf("%d", &num);
    itoa(num, str, 10); /* 对于较高版本的编译器, 请使用 _itoa */
    len = strlen(str);

    printf("\n该数的各位数之逆序为: ");
    for (i = 0; str[i] != '\0'; i++)
    {
        printf("%c", str[len - i - 1]);
        sum += str[i] - '0';
    }
    printf("\n该数的各位数之和为: %d\n", sum);

    return 0;
}
```

参考答案 2 (递归函数)：

```
#include <stdio.h>

void Invert(int num);
int Sum(int num);

int main(void)
{
    int num;

    printf("请输入一个数: ");
    scanf("%d", &num);

    printf("\n该数的各位数之逆序为: ");
```

```

    Invert(num);
    printf("该数的各位数之和为: %d\n", Sum(num));

    return 0;
}

void Invert(int num)
{
    if (num < 10)
    {
        printf("%d\n", num);
    }
    else
    {
        printf("%d", num % 10);
        Invert(num / 10);
    }
}

int Sum(int num)
{
    int result;

    if (num < 10)
    {
        result = num;
    }
    else
    {
        result = num % 10 + Sum(num / 10);
    }

    return result;
}

```

P796

DIFFICUTLY ★

在磁盘上新建一个文件“Test.txt”，将从键盘读入的多个字符存储到该文件中，要求如下：

1. 若输入的字符中有小写字母，则应先将其转换为大写后再存入。
2. 输入“!”表示输入结束且“!”不存入文件中。
3. 当文件创建失败或向文件写入字符时出错，应显示指定的出错信息并终止程序的执行。
4. 程序的返回值(即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值, 也称退出代码)规定为：
 - a) 运行正常，返回 0。
 - b) 文件创建失败，返回 1。
 - c) 向文件写入字符时出错，返回 2。

可用素材： printf("\nCreate file error!\n")
 printf("Input chars: ")
 printf("\nWriting file error!\n")

程序的运行效果应类似地如图 796.1 所示，文件“Test.txt”的内容应类似地如图 796.2 所示，金色部分是从键盘输入的内容。



图 796.1



图 796.2 生成的文件“Test.txt”之内容

参考答案:

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

```
int main(void)
```

```
{
```

```
    FILE *fp;
```

```
    int ch;
```

```
    fp = fopen("Test.txt", "w");
```

```
    if (fp == NULL)
```

```
    {
```

```
        printf("\nCreate file error!\n");
```

```
        return 1;
```

```
    }
```

```
    printf("Input chars: ");
```

```
    while (1)
```

```
    {
```

```
        ch = getchar();
```

```
        if (ch != EOF)
```

```
        {
```

```
            if (ch >= 'a' && ch <= 'z')
```

```
            {
```

```
                if (fputc(ch - 32, fp) == EOF)
```

```
                {
```

```
                    printf("\nWriting file error!\n");
```

```
                    fclose(fp);
```

```
                    return 2;
```

```
                }
```

```
            }
```

```
            else if (ch == '!')
```

```
            {
```

```
                break;
```

```
            }
```

```
            else
```

```
            {
```

```
                if (fputc(ch, fp) == EOF)
```

```
                {
```

```
                    printf("\nWriting file error!\n");
```

```
                    fclose(fp);
```

```
                    return 2;
```

```
                }
```

```
            }
```



```

    }
}
}
fclose(fp);
return 0;
}

```

P219

DIFFICUTLY ★★

从键盘读入一行字符（约定：字符数 ≤ 127 字节），将其中的数字字符以及这些数字字符的数量在屏幕上显示。

注：要求先显示这些数字字符的数量。

可用素材： `printf("Please input string: ")`
`printf("\nshu zi ge shu wei: ...`
`printf("\nshu zi wei: ...`

程序的运行效果应类似地如图 219.1 所示，金色部分是从键盘输入的内容。



图 219.1

参考答案：

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

```

int main(void)
{
    char str[128], seq[128];
    int i, j, num = 0;

    printf("Please input string: ");
    gets(str);

    for (i = 0, j = 0; str[i] != '\0'; i++)
    {
        if (str[i] >= '0' && str[i] <= '9')
        {
            num++;
            seq[j] = str[i];
            j++;
        }
    }
    seq[j] = '\0';
    printf("\nshu zi ge shu wei: %d", num);
    printf("\nshu zi wei: %s\n", seq);

    return 0;
}

```

P220

DIFFICUTLY ★★

从键盘读入一行字符（约定：字符数 ≤ 127 字节），统计及输出其中的字母、数字、空格和其他符号的个数。

可用素材： `printf("Please input string: ")`
`printf("\nzimu=...,shuzi=...,kongge=...,qita=...\n"...`

程序的运行效果应类似地如图 220.1 所示，金色部分是从键盘输入的内容。



图 220.1

参考答案：

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

```
int main(void)
```

```
{
```

```
    char str[128];
```

```
    int i, zimu = 0, shuzi = 0, kongge = 0, qita = 0;
```

```
    printf("Please input string: ");
```

```
    gets(str);
```

```
    for (i = 0; str[i] != '\0'; i++)
```

```
    {
```

```
        if (str[i] >= 48 && str[i] <= 57)
```

```
        {
```

```
            shuzi++;
```

```
        }
```

```
        else if (str[i] >= 65 && str[i] <= 90 || str[i] >= 97 && str[i] <= 122)
```

```
        {
```

```
            zimu++;
```

```
        }
```

```
        else if (str[i] == ' ')
```

```
        {
```

```
            kongge++;
```

```
        }
```

```
        else
```

```
        {
```

```
            qita++;
```

```
        }
```

```
    }
```

```
    printf("\nzimu=%d,shuzi=%d,kongge=%d,qita=%d\n", zimu, shuzi, kongge, qita);
```

```
    return 0;
```

```
}
```

P222

DIFFICUTLY ★★

从键盘上读入一行字符（约定：字符数 ≤ 127 字节），按以下方法将其加密变换：

A \rightarrow Z, B \rightarrow Y, C \rightarrow X, \cdots , Z \rightarrow A; a \rightarrow z, b \rightarrow y, c \rightarrow x, \cdots , z \rightarrow a。

即字母 A 变成 Z、字母 B 变成 Y、 \cdots ，非字母字符不变。最后在屏幕上先显示这一行字符的长度，再显示生成的密文。

可用素材： `printf("Please input string: ")`
`printf("\nzi fu chuan chang du: ...`
`printf("\nmi wen: ...`

程序的运行效果应类似地如图 222.1 所示，金色部分是从键盘输入的内容。

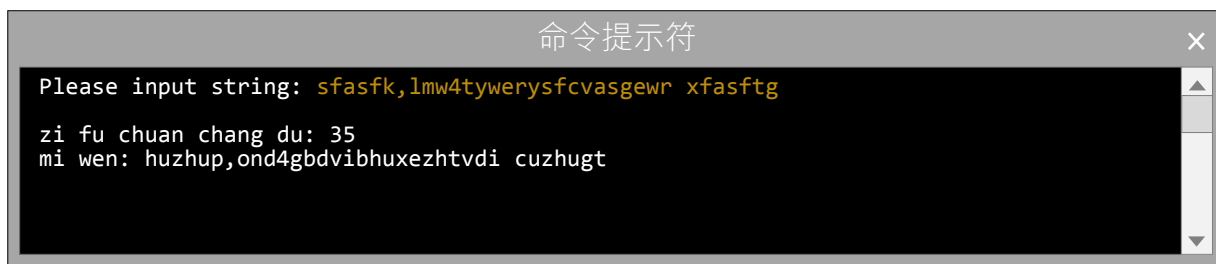


图 222.1

参考答案：

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

int main(void)
{
    char str[128];
    int len, i;

    printf("Please input string: ");
    gets(str);
    len = strlen(str);

    printf("\nzi fu chuan chang du: %d", len);
    for (i = 0; str[i] != '\0'; i++)
    {
        if (str[i] >= 'A' && str[i] <= 'Z')
        {
            str[i] = 'A' + 'Z' - str[i];
        }
        else if (str[i] >= 'a' && str[i] <= 'z')
        {
            str[i] = 'a' + 'z' - str[i];
        }
    }
    printf("\nmi wen: %s\n", str);

    return 0;
}
```

P717

DIFFICUTLY ★★

输入 10 个整型数存入一维数组，输出值和下标都为奇数（数组第 1 个元素的下标为 0）的元素及其个

数，要求先输出个数。

可用素材： `printf("Input 10 integers: ")`
`printf("\ncount=...\n"...`
`printf("a[...]=...\n"...`

程序的运行效果应类似地如图 717.1 所示，金色部分是从键盘输入的内容。

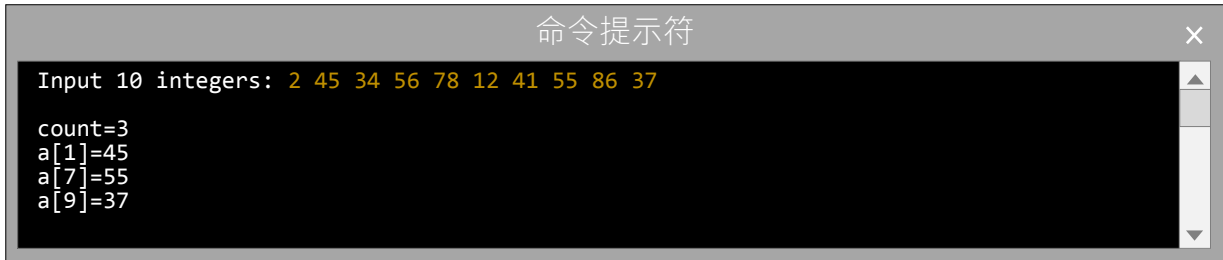


图 717.1

参考答案：

`#include <stdio.h>`

```
int main(void)
{
    int arr[10], i, count, odd = 1;

    printf("Input 10 integers: ");
    for (i = 0; i < 10; i++)
    {
        scanf("%d", &arr[i]);
    }

    for (i = 0; i < 10; i++)
    {
        if (i % 2 != 0 && arr[i] % 2 != 0)
        {
            count = odd++;
        }
    }
    printf("\ncount=%d\n", count);

    for (i = 0; i < 10; i++)
    {
        if (i % 2 != 0 && arr[i] % 2 != 0)
        {
            printf("a[%d]=%d\n", i, arr[i]);
        }
    }

    return 0;
}
```

P725

DIFFICUTLY ★★

利用数组，求斐波拉契数列的前 n (约定 $3 \leq n \leq 20$) 项并输出到屏幕上（数和数之间用水平制表符 '\t' 隔开），斐波拉契公式为： $f(1) = 1$, $f(2) = 1$, $f(n) = f(n-1) + f(n-2)$ ($n \geq 3$)。

可用素材： `printf("input a data(3--20): ")`

程序的运行效果应类似地如图 725.1 所示，金色部分是从键盘输入的内容。

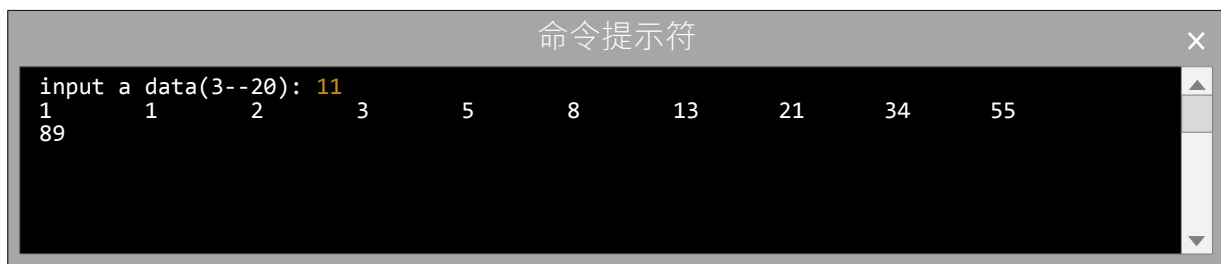


图 725.1

参考答案:

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

```
int main(void)
{
    int num, i, arr[20] = { 1, 1 };

    do
    {
        printf("input a data(3--20): ");
        scanf("%d", &num);
    } while (num < 3 || num > 20);

    for (i = 2; i < num; i++)
    {
        arr[i] = arr[i - 2] + arr[i - 1];
    }
    for (i = 0; i < num; i++)
    {
        if (i == num - 1)
        {
            printf("%d", arr[i]);
        }
        else
        {
            printf("%d\t", arr[i]);
        }
    }
    putchar('\n');

    return 0;
}
```

P211

DIFFICUTLY ★★

根据输入的 n 在屏幕上显示对应的以 “#” 组成的菱形图案。

可用素材: `printf("Please input n: ")`

程序的运行效果应类似地如图 211.1 和图 211.2 所示，金色部分是从键盘输入的内容。



图 211.1



图 211. 2

参考答案:

```
#include <stdio.h>

int main(void)
{
    int num, i, j;

    printf("Please input n: ");
    scanf("%d", &num);

    for (i = 1; i <= num + 1; i++)
    {
        for (j = i; j <= num; j++)
        {
            printf(" ");
        }
        for (j = 1; j <= (i * 2 - 1); j++)
        {
            printf("#");
        }
        putchar('\n');
    }
    for (i = 1; i <= num; i++)
    {
        for (j = 1; j <= i; j++)
        {
            printf(" ");
        }
        for (j = 1; j <= 2 * (num - i) + 1; j++)
        {
            printf("#");
        }
        putchar('\n');
    }

    return 0;
}
```

P212

DIFFICUTLY ★★

根据输入的 n (约定 $n > 1$) 在屏幕上显示对应的图案。

可用素材: `printf("Please input n: ")`

程序的运行效果应类似地如图 212. 1 和图 212. 2 所示, 金色部分是从键盘输入的内容。



图 212.1



图 212.2

参考答案 1 (常规) :

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

```
int main(void)
{
    int num, ch = 'a', i, j;

    do
    {
        printf("Please input n: ");
        scanf("%d", &num);
    } while (num < 1);

    for (i = 1; i <= num; i++)
    {
        for (j = i; j <= (num - 1); j++)
        {
            printf(" ");
        }
        printf("%c", ch + (i - 1));
        if (i > 1)
        {
            for (j = 1; j <= 2 * (i - 1) - 1; j++)
            {
                printf(" ");
            }
            printf("%c", ch + (i - 1));
        }
        putchar('\n');
    }
    for (i = num - 1; i >= 1; i--)
    {
        for (j = i; j <= num - 1; j++)
        {
            printf(" ");
        }
        printf("%c", ch + (i - 1));
    }
}
```

```

        if (i > 1)
        {
            for (j = 1; j <= 2 * (i - 1) - 1; j++)
            {
                printf(" ");
            }
            printf("%c", ch + (i - 1));
        }
        putchar('\n');
    }

    return 0;
}

```

参考答案 2 (直线方程) :

```

#include <stdio.h>

int main(void)
{
    int num, ch = 'a' - 1, i, j;

    printf("Please input n: ");
    scanf("%d", &num);

    for (i = 1; i <= 2 * num - 1; i++)
    {
        if (i <= num)
        {
            ch++;
        }
        else
        {
            ch--;
        }
        for (j = 1; j <= 2 * num - 1; j++)
        {
            if (i + j == num + 1 || i + j == 3 * num - 1 || i - j == 1 - num || i - j == num - 1)
            {
                printf("%c", ch);
            }
            else
            {
                printf(" ");
            }
        }
        putchar('\n');
    }

    return 0;
}

```

P213

DIFFICUTLY ★★

根据输入的 n (约定 $n > 0$) 在屏幕上显示对应的图案。

可用素材: `printf("Please input n: ")`

程序的运行效果应类似地如图 213.1 和图 213.2 所示, 金色部分是从键盘输入的内容。



图 213.1



图 213.2

参考答案:

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

```
int main(void)
```

```
{
```

```
    int num, i, j;
```

```
    do
```

```
    {
```

```
        printf("Please input n: ");
```

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

```
    } while (num < 0);
```

```
    for (i = 1; i <= num; i++)
```

```
    {
```

```
        for (j = 1; j <= (i - 1); j++)
```

```
        {
```

```
            printf(" ");
```

```
        }
```

```
        printf("@");
```

```
        if (i != num)
```

```
        {
```

```
            for (j = i; j <= 2 * (num - 1) - i; j++)
```

```
            {
```

```
                printf(" ");
```

```
            }
```

```
            printf("@");
```

```
        }
```

```
        putchar('\n');
```

```
    }
```

```
    for (i = num - 1; i >= 1; i--)
```

```
    {
```

```
        for (j = 1; j <= (i - 1); j++)
```

```
        {
```

```
            printf(" ");
```

```
        }
```

```
        printf("@");
```

```

        for (j = 1; j <= 2 * (num - i) - 1; j++)
        {
            printf(" ");
        }
        printf("@\n");
    }

    return 0;
}

```

P214

DIFFICUTLY ★★

根据输入的 n (约定 $n > 0$) 在屏幕上显示对应的图案。

可用素材: `printf("Please input n: ")`

程序的运行效果应类似地如图 214.1 和图 214.2 所示，金色部分是从键盘输入的内容。



图 214.1



图 214.2

参考答案:

```

#include <stdio.h>

int main(void)
{
    int num, i, j;

    do
    {
        printf("Please input n: ");
        scanf("%d", &num);
    } while (num < 0);

    for (i = 1; i <= num; i++)
    {
        printf("$");
        if (i != num)
        {

```

```

        for (j = 1; j <= num - i - 1; j++)
        {
            printf(" ");
        }
        printf("$\n");
    }
    if (i == num)
    {
        putchar('\n');
    }
}
for (i = num - 1; i >= 1; i--)
{
    printf("$");
    for (j = 1; j <= num - i - 1; j++)
    {
        printf(" ");
    }
    printf("$\n");
}

return 0;
}

```

P216

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的最大数及其所在的行列数, m ($2 \leq m \leq 20$) 及矩阵元素从键盘输入 (只考虑 int 型, 且不需考虑求和的结果可能超出 int 型能表示的范围)。

可用素材: `printf("Please input m: ")`
`printf("Please input array:\n")`
`printf("\nmax=...,i=...,j=...\n")...`

程序的运行效果应类似地如图 216.1 所示, 金色部分是从键盘输入的内容。



图 216.1

参考答案:

```

#include <stdio.h>

int main(void)
{
    int arr[20][20], m0, i, j, max, maxi = 0, maxj = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    } while (m0 < 2 || m0 > 20);
}

```

```

printf("Please input array:\n");
for (i = 0; i < m0; i++)
{
    for (j = 0; j < m0; j++)
    {
        scanf("%d", &arr[i][j]);
    }
}

for (i = 0, max = arr[0][0]; i < m0; i++)
{
    for (j = 0; j < m0; j++)
    {
        if (arr[i][j] > max)
        {
            max = arr[i][j];
            maxi = i;
            maxj = j;
        }
    }
}
printf("\nmax=%d,i=%d,j=%d\n", max, maxi, maxj);

return 0;
}

```

P217

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的对角线上元素之和， m ($2 \leq m \leq 20$) 及矩阵元素从键盘输入（只考虑 int 型，且不需考虑求和的结果可能超出 int 型能表示的范围）。

可用素材： printf("Please input m: ")
printf("Please input array:\n")
printf("\nsum=...\n")...

程序的运行效果应类似地如图 217.1 所示，金色部分是从键盘输入的内容。



图 217.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int arr[20][20], m0, i, j, sum = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    }
}

```

```

} while (m0 < 2 || m0 > 20);

printf("Please input array:\n");
for (i = 0; i < m0; i++)
{
    for (j = 0; j < m0; j++)
    {
        scanf("%d", &arr[i][j]);
    }
}

for (i = 0; i < m0; i++)
{
    for (j = 0; j < m0; j++)
    {
        if (i != j)
        {
            if (i + j == m0 - 1)
            {
                sum += arr[i][j];
            }
        }
        else
        {
            sum += arr[i][j];
        }
    }
}
printf("\nsum=%d\n", sum);

return 0;
}

```

P218

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的周边元素之和， m ($2 \leq m \leq 20$) 及矩阵元素从键盘输入（只考虑 int 型，且不需考虑求和的结果可能超出 int 型能表示的范围）。

可用素材： printf("Please input m: ")
 printf("Please input array:\n")
 printf("\nsum=...\n")...

程序的运行效果应类似地如图 218.1 所示，金色部分是从键盘输入的内容。



图 218.1

参考答案：

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

```
int main(void)
```

```

{
    int arr[20][20], m0, i, j, sum = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    } while (m0 < 2 || m0 > 20);

    printf("Please input array:\n");
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }

    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            if (0 == i || 0 == j || m0 - 1 == i || m0 - 1 == j)
            {
                sum += arr[i][j];
            }
        }
    }
    printf("\nsum=%d\n", sum);

    return 0;
}

```

P234

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的第 0 行和最后一行所有数之和, m ($2 \leq m \leq 20$) 及矩阵元素从键盘输入 (只考虑 int 型)。

可用素材: `printf("Please input m: ")`
`printf("Please input array:\n")`
`printf("\nSum=...\n")...`

程序的运行效果应类似地如图 234.1 所示, 金色部分是从键盘输入的内容。



图 234.1

参考答案:

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

```
int main(void)
```

```

{
    int arr[20][20], m0, i, j, sum = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    } while (m0 < 2 || m0 > 20);

    printf("Please input array:\n");
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }

    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            if (i == 0 || i == m0 - 1)
            {
                sum += arr[i][j];
            }
        }
    }
    printf("\nSum = %d\n", sum);

    return 0;
}

```

P235

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的第 0 列和最后一列所有数之和， m ($2 \leq m \leq 20$) 及矩阵元素从键盘输入（只考虑 int 型）。

可用素材： printf("Please input m: ")
 printf("Please input array:\n")
 printf("\nSum=...\n")...

程序的运行效果应类似地如图 235.1 所示，金色部分是从键盘输入的内容。



图 235.1

参考答案：

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

```
int main(void)
```

```

{
    int arr[20][20], m0, i, j, sum = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    } while (m0 < 2 || m0 > 20);

    printf("Please input array:\n");
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }

    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            if (j == 0 || j == m0 - 1)
            {
                sum += arr[i][j];
            }
        }
    }
    printf("\nSum = %d\n", sum);

    return 0;
}

```

P236

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的第 0 行、第 2 行和最后一行所有数之和， m ($4 \leq m \leq 20$) 及矩阵元素从键盘输入（只考虑 int 型）。

可用素材： `printf("Please input m: ")`
`printf("\nPlease input array:\n")`
`printf("\nSum = ...\n")`

程序的运行效果应类似地如图 236.1 所示，金色部分是从键盘输入的内容。



图 236.1

参考答案：

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

```
int main(void)
```



```

{
    int arr[20][20], m0, i, j, sum = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    } while (m0 < 4 || m0 > 20);

    printf("Please input array:\n");
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            scanf("%d", &arr[i][j]);
            if (i == 0 || i == 2 || i == m0 - 1)
            {
                sum += arr[i][j];
            }
        }
    }
    printf("\nSum = %d\n", sum);

    return 0;
}

```

P237

DIFFICUTLY ★★

求任意的一个 $m \times m$ 矩阵的第 0 列、第 2 列和最后一列所有数之和， m ($4 \leq m \leq 20$) 及矩阵元素从键盘输入（只考虑 int 型）。

可用素材： printf("Please input m: ")
 printf("\nPlease input array:\n")
 printf("\nSum = ...\n")

程序的运行效果应类似地如图 237.1 所示，金色部分是从键盘输入的内容。



图 237.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int arr[20][20], m0, i, j, sum = 0;

    do
    {
        printf("Please input m: ");
        scanf("%d", &m0);
    }

```

```

    } while (m0 < 4 || m0 > 20);

    printf("Please input array:\n");
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < m0; j++)
        {
            scanf("%d", &arr[i][j]);
            if (j == 0 || j == 2 || j == m0 - 1)
            {
                sum += arr[i][j];
            }
        }
    }
    printf("\nSum = %d\n", sum);

    return 0;
}

```

P724

DIFFICUTLY ★★

从键盘输入 1~9 之间的一个数，根据输入的数，输出对应的下三角乘法口诀表。要求积的输出占 3 个宽度，且左对齐。

可用素材： `printf("input a number(1~9): ")`

程序的运行效果应类似地如图 724.1 和图 724.2 所示，金色部分是从键盘输入的内容。

图 724.1

图 724.2

参考答案：

```

#include <stdio.h>

int main(void)
{
    int num, i, j;

    printf("input a number(1~9): ");
    scanf("%d", &num);

```

```

    for (i = 1; i <= num; i++)
    {
        for (j = 1; j <= i; j++)
        {
            printf("%d*%d=%-3d", i, j, i * j);
        }
        putchar('\n');
    }

    return 0;
}

```

P739

DIFFICUTLY ★★

从键盘输入年月日日期，计算出该日期是公元纪年以来的第几天。

注：判断年份是否为闰年的方法——为 400 的倍数为闰年，如 2000 年；若非 100 的倍数，而是 4 的倍数，为闰年，如 1996 年。

可用素材： `printf("input a data(year-month-day):")`
`printf("\nThe result is\n")...`

程序的运行效果应类似地如图 739.1 所示，金色部分是从键盘输入的内容。



图 739.1

参考答案：

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

```
int main(void)
```

```
{
```

```
    int year, month, day, i;
```

```
    int total[13] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
```

```
    double sumyear = 0, summonth = 0, sum;
```

```
    printf("input a data(year-month-day):");
```

```
    scanf("%d-%d-%d", &year, &month, &day);
```

```
    for (i = 1; i < year; i++)
```

```
    {
```

```
        if (i % 400 == 0 || i % 100 != 0 && i % 4 == 0)
```

```
        {
```

```
            sumyear += 366;
```

```
        }
```

```
        else
```

```
        {
```

```
            sumyear += 365;
```

```
        }
```

```
    }
```

```
    for (i = 0; i < month; i++)
```

```
    {
```

```
        summonth += total[i];
```

```
    }
```

```

    if (year % 400 == 0 && month > 2 || year % 100 != 0 && year % 4 == 0 && month > 2)
    {
        summonth += 1;
    }
    sum = sumyear + summonth + day;
    printf("The result is %.01f.\n", sum);

    return 0;
}

```

P740

DIFFICUTLY ★★

公鸡每只 5 元，母鸡每只 3 元，小鸡每 3 只 1 元，用 M 元钱买 N 只鸡，问公鸡、母鸡、小鸡各几只。

可用素材：

```

printf("Input the money: ")
printf("Input the number: ")
printf("\n cock   hen chick\n")

```

程序的运行效果应类似地如图 740.1 和图 740.2 所示，金色部分是从键盘输入的内容。

```

命令提示符
Input the money: 100
Input the number: 100

cock   hen chick
0      25    75
4      18    78
8      11    81
12     4     84

```

图 740.1

```

命令提示符
Input the money: 80
Input the number: 90

cock   hen chick
1      17    72
5      10    75
9       3    78

```

图 740.2

参考答案：

```

#include <stdio.h>

int main(void)
{
    int money, number, cock, hen, chick;

    printf("Input the money: ");
    scanf("%d", &money);
    printf("Input the number: ");
    scanf("%d", &number);

    printf("\n cock   hen chick\n");
    for (cock = 0; cock <= number; cock++)
    {
        for (hen = 0; hen <= number; hen++)
        {
            for (chick = 0; chick <= number; chick++)

```

```

    {
        if (cock * 5 + hen * 3 + chick / 3 == money && chick % 3 == 0 &&
            cock + hen + chick == number)
        {
            printf("%6d%6d%6d\n", cock, hen, chick);
        }
    }
}

return 0;
}

```

P742

DIFFICUTLY ★★

现有两个一维数组（各含 5 个整型元素）设为 **A**、**B**，从键盘分别输入数据给这两个数组。计算 **A** 数组正序位置与 **B** 数组逆序对应位置积的和。

可用素材： `printf("Input A: ")`
`printf("Input B: ")`
`printf("\nsum=...\n")...`

程序的运行效果应类似地如图 742.1 所示，金色部分是从键盘输入的内容。



图 742.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int arrA[5], arrB[5], i, j, sum = 0;

    printf("Input A: ");
    for (i = 0; i < 5; i++)
    {
        scanf("%d", &arrA[i]);
    }
    printf("Input B: ");
    for (i = 0; i < 5; i++)
    {
        scanf("%d", &arrB[i]);
    }

    for (i = 0, j = 4; i < 5; i++, j--)
    {
        sum += arrA[i] * arrB[j];
    }
    printf("\nsum=%d\n", sum);

    return 0;
}

```

P748

DIFFICUTLY ★★

从键盘上输入矩阵的阶数 n ($n \leq 14$)，矩阵中元素的值等于其位置的行数和列数之和的 n 倍（行列的值从 0 开始计数），如 $n = 3$ 时，矩阵为 $\begin{pmatrix} 0 & 3 & 6 \\ 3 & 6 & 9 \\ 6 & 9 & 12 \end{pmatrix}$ 。先输出该矩阵（显示时每个数宽度为 4、右对齐），然后计算输出 sum_1 和 sum_2 的值： sum_1 为矩阵中所有不靠边元素之和， sum_2 为矩阵的一条对角线元素之和。

可用素材： `printf("Enter n: ")`
`printf("\nsum1=...\nsum2=...\n"...`

程序的运行效果应类似地如图 748.1 所示，金色部分是从键盘输入的内容。



图 748.1

参考答案：

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

```
int main(void)
{
    int arr[14][14], num, i, j, sum1 = 0, sum2 = 0;

    do
    {
        printf("Enter n: ");
        scanf("%d", &num);
    } while (num < 0 || num > 14);

    for (i = 0; i < num; i++)
    {
        for (j = 0; j < num; j++)
        {
            arr[i][j] = (i + j) * num;
            printf("%4d", arr[i][j]);
        }
        putchar('\n');
    }

    for (i = 1; i < num - 1; i++)
    {
        for (j = 1; j < num - 1; j++)
        {
            sum1 += arr[i][j];
        }
    }

    for (i = 0; i < num; i++)
    {
        for (j = 0; j < num; j++)
        {
            if (i == j)
            {
                sum2 += arr[i][j];
            }
        }
    }
}
```

```

    }
}
printf("\nsum1=%d\nsum2=%d\n", sum1, sum2);

return 0;
}

```

P770

DIFFICUTLY ★★

实现十进制转 num 换成 R 进制数, num 和 R 从键盘读入 (假定 num 为 `int` 且 $-32767 \leq num \leq 32767$, R 为 `int` 且 $2 \leq R \leq 16$)。

注: 程序中不能使用库函数 `itoa()`、`ltoa()`、`ultoa()` 或使用同名的变量、函数、单词。

可用素材: `printf("input the num, R: ")`
`printf("Result: \n")`

程序的运行效果应类似地如图 770.1 和图 770.2 所示, 金色部分是从键盘输入的内容。



图 770.1



图 770.2

参考答案:

```

#include <stdio.h>
#include <string.h>

void Fn(int num, int R0, char data[]);

int main(void)
{
    int num, R0, i, len;
    char data[100];

    do
    {
        printf("input the num, R: ");
        scanf("%d,%d", &num, &R0);
    } while (R0 < 2 || R0 > 16);

    Fn(num, R0, data);
    len = strlen(data);

```

```

    printf("Result: \n");
    if (num < 0)
    {
        printf("-");
    }
    for (i = len - 1; i >= 0; i--)
    {
        printf("%c", data[i]);
    }
    if (num == 0)
    {
        printf("0");
    }
    putchar('\n');

    return 0;
}

void Fn(int num, int R0, char data[])
{
    int i, mod;

    if (num < 0)
    {
        num *= -1;
    }

    for (i = 0; num != 0; i++)
    {
        mod = num % R0;
        if (mod >= 10)
        {
            mod -= 10;
            mod += 'A';
        }
        else
        {
            mod += '0';
        }
        data[i] = mod;
        num /= R0;
    }
    data[i] = '\0';
}

```

P818

DIFFICUTLY ★★

程序 P818.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：找出任意的一个 $m \times n$ 矩阵每一行上的最大值的列下标并按示例格式要求显示。 m 、 n ($2 \leq m \leq 20$, $2 \leq n \leq 20$) 及矩阵元素从键盘输入。

可用素材： `printf("The max value in line %d is %d\n"...`

程序的运行效果应类似地如图 818.1 所示，金色部分是从键盘输入的内容。

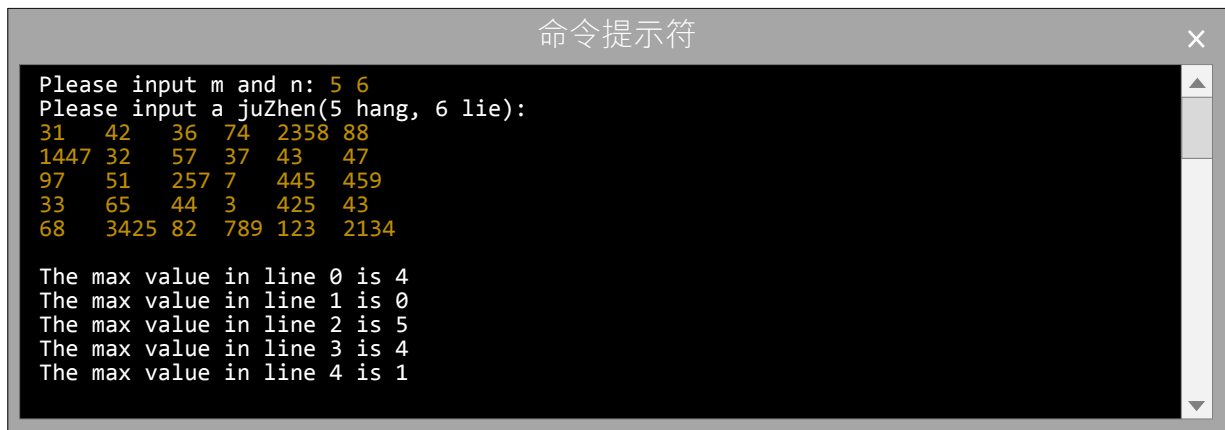


图 818.1

附件:  P818.c

文件内容:

P818.c

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

```
#define N 20
```

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int m, n;
```

```
    int hang, lie, juZhen[N][N];
```

```
    printf("Please input m and n: ");
```

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

```
    printf("Please input a juZhen(%d hang, %d lie):\n", m, n);
```

```
    for (hang = 0; hang < m; hang++)
```

```
    {
```

```
        for (lie = 0; lie < n; lie++)
```

```
        {
```

```
            scanf("%d", &juZhen[hang][lie]);
```

```
        }
```

```
    }
```

```
    puts("");
```

```
<B>
```

```
/* userCode(<50字符): 调用函数找出每一行上的最大值的列下标并按示例格式
```

```
要求显示 */
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<C>
```

参考答案:

```
<A> void GetMaxPol(int arr[][N], int hang, int lie);
```

```
<B> GetMaxPol(juZhen, hang, lie);
```

```
<C> void GetMaxPol(int arr[][N], int hang, int lie)
```

```
{
```

```
    int i, j, max, maxpol;
```

```

for (i = 0; i < hang; i++)
{
    max = arr[i][0];
    maxpol = 0;
    for (j = 0; j < lie; j++)
    {
        if (arr[i][j] > max)
        {
            max = arr[i][j];
            maxpol = j;
        }
    }
    printf("The max value in line %d is %d\n", i, maxpol);
}
}

```

P823

DIFFICUTLY ★★

程序 P823.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。

程序的功能是：将 m ($2 \leq m \leq 20$) 行 m 列的二维数组 **arrayA** 中的最后一行放到二维数组 **arrayB** 的第 0 列中，把二维数组 **arrayA** 中的第 0 行放到二维数组 **arrayB** 的最后一列中，二维数组 **arrayB** 中的其他数据和 **arrayA** 一致。

程序的运行效果应类似地如图 823.1 所示，金色部分是从键盘输入的内容。



图 823.1

附件:  P823.c

文件内容:

P823.c

```

#include <stdio.h>

#define MAX 20

/* userCode(<80字符>): 自定义函数之原型声明 */
<A>

int main(void)
{
    int arrayA[MAX][MAX], arrayB[MAX][MAX], i, j, m;

```

```

printf("Please input m: ");
scanf("%d", &m);

printf("Please input arrayA:\n");
for (i=0; i<m; i++)
{
    for (j=0; j<m; j++)
    {
        scanf("%d", &arrayA[i][j]);
    }
}

<B> /* userCode(<60字符): 调用函数实现数组内容变换 */
printf("\nafter rotate:\n");
for (i=0; i<m; i++)
{
    for (j=0; j<m; j++)
    {
        printf("%6d", arrayB[i][j]);
    }
    putchar('\n');
}

return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<C>

```

参考答案:

```

<A> void Change(int arr1[][MAX], int arr2[][MAX], int m);
<B> Change(arrayA, arrayB, m);
<C> void Change(int arr1[][MAX], int arr2[][MAX], int m)
{
    int i, j;

    for (i = 0; i < m; i++)
    {
        for (j = 0; j < m; j++)
        {
            arr2[i][j] = arr1[i][j];
        }
    }
    for (j = 0; j < m; j++)
    {
        arr2[j][0] = arr1[m - 1][j];
    }
    for (j = 0; j < m; j++)
    {
        arr2[j][m - 1] = arr1[0][j];
    }
}

```

P311
DIFFICUTLY ★★

编写一程序实现以下功能:

1. 从键盘上先后读入两个字符串，假定存储在字符数组 s_1 和 s_2 中。注意，这两个字符串最长均可达到 127 个字符、最短均可为 0 个字符。
2. 将字符串 s_2 插入字符串 s_1 中，插入方法为： s_2 的第 i 个字符插入到原 s_1 的第 i 个字符后，如果 s_2 比 s_1 （假定 s_1 的长度为 L_1 ）长，则 s_2 的第 L_1 个字符开始到 s_2 结尾的所有字符按在 s_2 中的顺序放在新生成的 s_1 后。例如， s_1 输入为“123456789”， s_2 输入为“abcdefghijk”，则输出的 s_1 为“1a2b3c4d5e6f7g8h9ijk”。
3. 在屏幕上输出新生成的 s_1 。

可用素材： `printf("Please input string1:");`
`printf("Please input string2:");`
`printf("\nstring1:...\\n"...`

提示： 合并时可使用中间数组。

程序的运行效果应类似地如图 311.1 所示，金色部分是从键盘输入的内容。



图 311.1

参考答案：

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

int main(void)
{
    char str1[256], str2[128], str3[256];
    int i, j;

    printf("Please input string1:");
    gets(str1);
    printf("Please input string2:");
    gets(str2);

    for (i = 0, j = 0; str1[i] != '\0' && str2[i] != '\0'; i++, j += 2)
    {
        str3[j] = str1[i];
        str3[j + 1] = str2[i];
    }

    if (strlen(str1) > strlen(str2))
    {
        for (; str1[i] != '\0'; i++, j++)
        {
            str3[j] = str1[i];
        }
    }
    else
    {
        for (; str2[i] != '\0'; i++, j++)
        {
            str3[j] = str2[i];
        }
    }
    str3[j] = '\0';
}
```

```

strcpy(str1, str3);
printf("\nstring1:%s\n", str1);

return 0;
}

```

P701

DIFFICUTLY ★★

从键盘读入 10 个数存放在一个数组中, 要求用户由小到大输入。如果发现读入的某个数不是由小到大, 则该数输入无效, 继续读入后面的输入数据。再将这 10 个数依次输出到屏幕上, 要求每个数的输出宽度为 5、不足 5 位的在其左边补上空格、数与数之间使用逗号“,”分隔。然后输入一个数, 要求找出该数是数组中第几个 (序号从 1 开始计算) 元素的值, 如果该数不在数组中, 则输出 “Not Find!”。

可用素材:

```

printf("please input 10 numbers: ")
printf("input the num to look for: ")
printf("\nthe position of ... is ...\n"...
printf("\nNot Find!\n")

```

程序的运行效果应类似地如图 701.1、图 701.2 和图 701.3 所示, 金色部分是从键盘输入的内容。

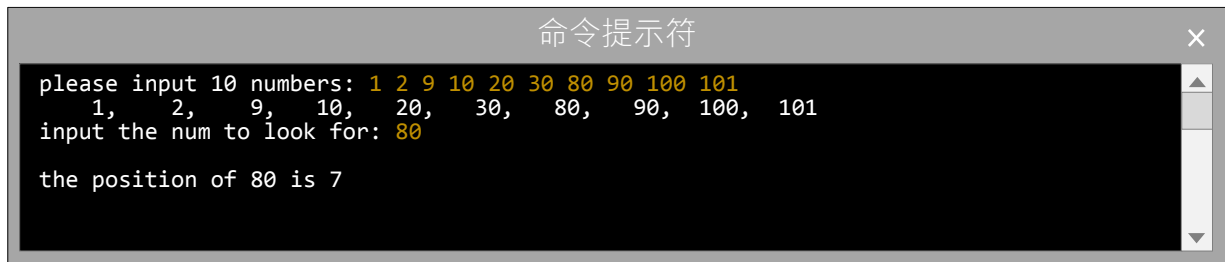


图 701.1

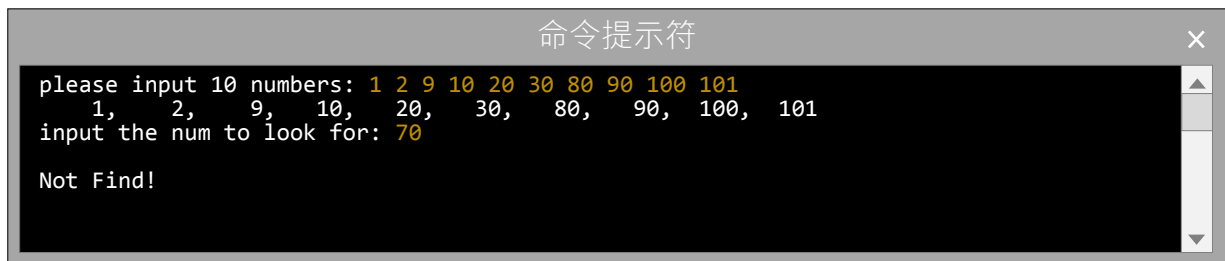


图 701.2



图 701.3

参考答案:

```

#include <stdio.h>

int main(void)
{
    int arr[10], num, i, flag = 0;

```

```

printf("please input 10 numbers: ");
scanf("%d", &arr[0]);
for (i = 1; i < 10; )
{
    scanf("%d", &arr[i]);
    if (arr[i] > arr[i - 1])
    {
        i++;
    }
}

for (i = 0; i < 10; i++)
{
    printf("%5d", arr[i]);
    if (i < 9)
    {
        printf(",");
    }
    else
    {
        putchar('\n');
    }
}

printf("input the num to look for: ");
scanf("%d", &num);
for (i = 0; i < 10; i++)
{
    if (arr[i] == num)
    {
        printf("\nthe position of %d is %d\n", num, i + 1);
        flag = 1;
        break;
    }
}
if (flag == 0)
{
    printf("\nNot Find!\n");
}

return 0;
}

```

P704

DIFFICUTLY ★★

一个数如果恰好等于它的因子之和，这个数就称为“完数”。例如 $6 = 1 + 2 + 3$ 。从键盘输入一个正整数（约定该数 ≤ 32767 、此时因子数 ≤ 100 ），找出该数以内的所有完数及其因子。

可用素材： printf("Please input an integer: ")
 printf("... is a wanshu"...)
 printf(" %d"...)

程序的运行效果应类似地如图 704.1 所示，金色部分是从键盘输入的内容。



图 704.1

参考答案:

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

```
int main(void)
{
    int num, i, j, k, count, arr[100], sum;

    do
    {
        printf("Please input an integer: ");
        scanf("%d", &num);
    } while (num < 0 || num > 32767);

    for (i = 1; i <= num; i++)
    {
        for (j = 1, count = 0; j < i; j++)
        {
            if (i % j == 0)
            {
                arr[count] = j;
                count++;
            }
        }
        for (k = 0, sum = 0; k < count; k++)
        {
            sum += arr[k];
        }
        if (i == sum)
        {
            printf("%6d is a wanshu", i);
            for (k = 0; k < count; k++)
            {
                printf(" %d", arr[k]);
            }
            putchar('\n');
        }
    }

    return 0;
}
```

P705

DIFFICUTLY ★★

从键盘输入星期几的第一个字母（接收字符请用 `getchar()`），由程序判断是星期几，并显示其对应的英文单词，如果第一个字母一样，则继续判断第二个字母。星期的第一个字母都为大写。如果输入的字符不能构成星期的英文单词，则输出“data error”。程序可重复判断，直至输入字母“Y”。

注：星期一至星期日对应的单词为 Monday、Tuesday、Wednesday、Thursday、Friday、Saturday、Sunday。

可用素材: `printf("Please input the letter of someday: ")`
`printf("Monday\n") printf("Wednesday\n") printf("Friday\n")`
`printf("Tuesday\n") printf("Thursday\n") printf("Saturday\n")`
`printf("Sunday\n") printf("data error\n")`

程序的运行效果应类似地如图 705.1 所示, 金色部分是从键盘输入的内容。

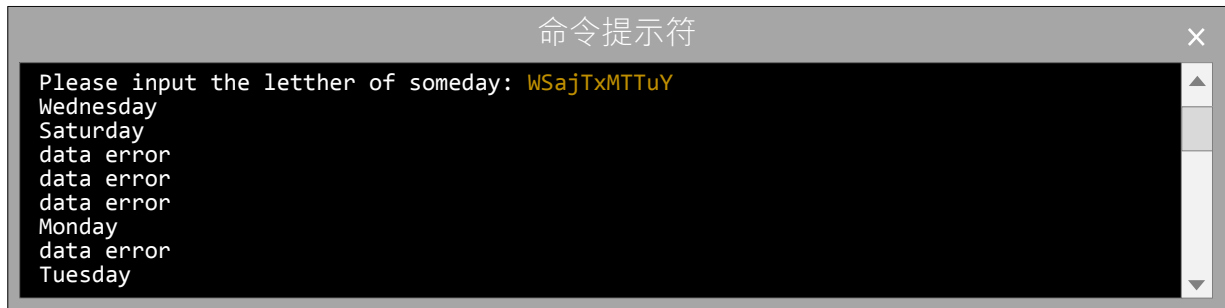


图 705.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    char str[1000], ch;
    int i, count = 0;

    printf("Please input the letter of someday: ");
    for (i = 0; i < 1000; i++)
    {
        ch = getchar();
        if (ch == 'Y' || ch == '\n')
        {
            break;
        }
        else
        {
            str[i] = ch;
            count++;
        }
    }

    for (i = 0; i < count; i++)
    {
        switch (str[i])
        {
            case 'M':
                printf("Monday\n");
                break;
            case 'W':
                printf("Wednesday\n");
                break;
            case 'F':
                printf("Friday\n");
                break;
            case 'T':
                if (str[i + 1] == 'u')
                {
                    printf("Tuesday\n");
                    i++;
                }
                else if (str[i + 1] == 'h')
                {

```



```

        printf("Thursday\n");
        i++;
    }
    else
    {
        printf("data error\n");
    }
    break;
case 'S':
    if (str[i + 1] == 'a')
    {
        printf("Saturday\n");
        i++;
    }
    else if (str[i + 1] == 'u')
    {
        printf("Sunday\n");
        i++;
    }
    else
    {
        printf("data error\n");
    }
    break;
default:
    printf("data error\n");
}
}

return 0;
}

```

P715

DIFFICUTLY ★★

如果一个正整数恰好等于它的所有因子（包括1但不包括自身）之和，则称之为“完数”，例如6的因子是1、2、3，且 $6 = 1 + 2 + 3$ ，因此6是完数。编写程序，输入数据范围（约定数的上限 ≤ 32767 、此时因子数 ≤ 100 ），输出该范围之内所有完数及其个数。

可用素材： `printf("Input 2 integer to determine the range: ")`
`printf("\nwanShu is: ")`
`printf("\n Count = ...\n")...`

程序的运行效果应类似地如图 715.1 所示，金色部分是从键盘输入的内容。



图 715.1

参考答案：

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

```
int main(void)
{
```

```

int min, max, i, j, k, count, arr[100], wan[100], countwan = 0, sum;

do
{
    printf("Input 2 integer to determine the range: ");
    scanf("%d%d", &min, &max);
} while (min < 0 || min > 32767 || max < 0 || max > 32767);

for (i = min; i <= max; i++)
{
    for (j = 1, count = 0; j < i; j++)
    {
        if (i % j == 0)
        {
            arr[count] = j;
            count++;
        }
    }

    for (k = 0, sum = 0; k < count; k++)
    {
        sum += arr[k];
    }
    if (i == sum)
    {
        wan[countwan] = i;
        countwan++;
    }
}

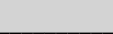
printf("\nwanShu is:");
for (i = 0; i < countwan; i++)
{
    printf(" %d", wan[i]);
}
printf("\n Count = %d\n", countwan);

return 0;
}

```

P768

DIFFICUTLY ★★

程序 P768.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：将从键盘读入的一个十六进制字符串转换成一个十进制数后输出。注意：

1. 可以只考虑转换成的十进制大小不会超过 long int 型所能表示的最大数。
2. 只需处理正数，不处理负数。
3. 输入的字母可以是大写也可以小写。
4. 程序中不能使用库函数 isxdigit()、sscanf() 或使用同名的变量、函数、单词。

程序的运行效果应类似地如图 768.1、图 768.2 和图 768.3 所示，**金色部分**是从键盘输入的内容。




图 768.1



图 768.2



图 768.3

附件:  P768.c

文件内容:

P768.c

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

```
/* User Code Begin(考生可在本行后添加代码, 例如全局变量的定义、函数原型声明等, 行数不限) */
```

```
<A>
```

```
/* User Code End(考生添加代码结束) */
```

```
int main(void)
```

```
{
```

```
    int flag; /* 标志输入数据是否合法, 0表示不合法 */
```

```
    long int result10; /* 转换结果 */
```

```
    char str[20];
```

```
    printf("input a data:");
```

```
    gets(str);
```

```
/* User Code Begin(考生可在本行后添加代码, 行数不限) */
```

```
<B>
```

```
/* User Code End(考生添加代码结束) */
```

```
    if (0 == flag)
```

```
    {
```

```
        printf("\ndata is error.\n");
```

```

    }
    else
    {
        printf("\nThe result is: %ld\n", result10);
    }

    return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分, 行数不限) */
<C>

```

参考答案:

```

<A> #include <string.h>
    #include <math.h>

    long GetD(char str[], int *pflag);
<B> result10 = GetD(str, &flag);
<C> long GetD(char str[], int *pflag)
    {
        int i, n0 = 0, len;
        long result = 0;

        for (i = 0; str[i] != '\0'; i++)
        {
            if (str[i] >= '0' && str[i] <= '9')
            {
                *pflag = 1;
            }
            else if (str[i] >= 'A' && str[i] <= 'F' || str[i] >= 'a' && str[i] <= 'f')
            {
                *pflag = 1;
            }
            else
            {
                *pflag = 0;
                break;
            }
        }

        if (*pflag == 1)
        {
            len = strlen(str);
            for (i = len - 1; i >= 0; i--)
            {
                if (str[i] >= '0' && str[i] <= '9')
                {
                    result += (long)((str[i] - '0') * pow(16, n0));
                    n0++;
                }
                else if (str[i] >= 'A' && str[i] <= 'F')
                {
                    result += (long)((str[i] - 'A' + 10) * pow(16, n0));
                    n0++;
                }
                else if (str[i] >= 'a' && str[i] <= 'f')
                {
                    result += (long)((str[i] - 'a' + 10) * pow(16, n0));
                    n0++;
                }
            }
        }
    }

```

```

    }
}

return result;
}

```

P771

DIFFICUTLY ★★

编程在一个已知的字符串（约定：字符数 ≤ 127 字节）中查找最长单词，输出最长单词的长度。

注意事项：

1. 只考虑输入的字符串中仅含空格和其它可见字符，仅用空格用来分隔不同单词。
2. 字符串中可能只有 1 个单词。
3. 字符串中可能 1 个单词都没有，此时最长单词的长度为 0。

可用素材： `printf("please input a string:\n")`
`printf("\nmax_length of the string is: %d\n"...`

程序的运行效果应类似地如图 771.1 所示，金色部分是从键盘输入的内容。



图 771.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    char str[128];
    int i, count = 0, max = 0;

    printf("please input a string:\n");
    gets(str);

    for (i = 0; str[i] != '\0'; i++)
    {
        if (str[i] != ' ')
        {
            count++;
        }
        else
        {
            if (count > max)
            {
                max = count;
            }
            count = 0;
        }
    }
    if (count > max)
    {
        max = count;
    }
}

```

```

    printf("\nmax_length of the string is: %d\n", max);

    return 0;
}

```

P773

DIFFICUTLY ★★

计算两个日期之间的天数。

注：判断年份是否为闰年的方法——为 400 的倍数为闰年，如 2000 年；若非 100 的倍数，而是 4 的倍数，为闰年，如 1996 年。

可用素材： `printf("from the date (****/**/)**):\n")`
`printf("to the date(****/**/)**):\n")`
`printf("\nsum=%d\n"...`

程序的运行效果应类似地如图 773.1 和图 773.2 所示，金色部分是从键盘输入的内容。



图 773.1



图 773.2

参考答案：

```

#include <stdio.h>
#include <math.h>

int Leap(int year);
int Calculate(int year, int month, int day);

int main(void)
{
    int sum, year1, month1, day1, year2, month2, day2;

    printf("from the date (****/**/)**):\n");
    scanf("%d/%d/%d", &year1, &month1, &day1);
    printf("to the date (****/**/)**):\n");
    scanf("%d/%d/%d", &year2, &month2, &day2);

    sum = abs(Calculate(year2, month2, day2) - Calculate(year1, month1, day1));
    printf("\nsum=%d\n", sum);

    return 0;
}

```

```

int Leap(int year)
{
    if (year % 400 == 0 || year % 4 == 0 && year % 100 != 0)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}

int Calculate(int year, int month, int day)
{
    int i, sum = 0;
    int mon[13] = { 0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

    for (i = 1; i < year; i++)
    {
        if (Leap(i) == 1)
        {
            sum += 366;
        }
        else
        {
            sum += 365;
        }
    }
    for (i = 1; i <= month - 1; i++)
    {
        if (Leap(year) == 1)
        {
            if (i == 2)
            {
                sum += mon[i] + 1;
            }
            else
            {
                sum += mon[i];
            }
        }
        else
        {
            sum += mon[i];
        }
    }
    sum += day;

    return sum;
}

```

P776

DIFFICUTLY ★★

n ($3 < n < 200$, 从键盘读入) 个学生围成一圈报数 (本身从 1 到 n 顺序编学号), 从第一个人开始顺序报 1, 2, 3, 1, 2, 3, ...; 凡报到 3 的人退出圈子, 要求根据输入 n 的值, 输出每次退出的学生的学号, 并且找出最后一个留在圈子中的学生的学号。

可用素材: `printf("Please input n: ")`
`printf("\nResult is:\n")`

```
printf("delete %d student: %d.\n"...  
printf("The remained student is %d.\n"...
```

程序的运行效果应类似地如图 776.1 所示，金色部分是从键盘输入的内容。



图 776.1

参考答案:

```
#include <stdio.h>  
#include <malloc.h>  
  
typedef struct Node  
{  
    int data;  
    struct Node *next;  
} JosephusNode;  
  
int Remain(int n);  
  
int main(void)  
{  
    int num;  
  
    do  
    {  
        printf("Please input n: ");  
        scanf("%d", &num);  
    } while (num < 3 || num > 200);  
  
    printf("\nResult is:\n");  
    printf("The remained student is %d.\n", Remain(num));  
  
    return 0;  
}  
  
int Remain(int num)  
{  
    int i, j;  
    JosephusNode *head, *tail;  
  
    head = tail = (JosephusNode *)malloc(sizeof(JosephusNode));  
    for (i = 1; i < num; i++)  
    {  
        tail->data = i;  
        tail->next = (JosephusNode *)malloc(sizeof(JosephusNode));  
        tail = tail->next;  
    }  
    tail->data = i;  
    tail->next = head;  
  
    for (i = 1; tail != head; i++)
```



```

{
    for (j = 1; j < 3; j++)
    {
        tail = head;
        head = head->next;
    }
    tail->next = head->next;
    printf("delete %d student: %d.\n", i, head->data);
    free(head);
    head = tail->next;
}
i = head->data;
free(head);

return i;
}

```

P808

DIFFICUTLY ★★

程序 P808.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：

1. 从键盘上先后读入两个字符串，假定存储在字符数组 **str₁** 和 **str₂** 中。注意，这两个字符串最长均可达到 127 个字符、最短均可 0 个字符。
2. 将字符串 **str₂** 插入字符串 **str₁** 中，插入方法为：**str₂** 的第 *i* 个字符插入到原 **str₁** 的第 *i* 个字符后，如果 **str₂** 比 **str₁**（假定 **str₁** 的长度为 *L₁*）长，则 **str₂** 的第 *L₁* 个字符开始到 **str₂** 结尾的所有字符按在 **str₂** 中的顺序放在新生成的 **str₁** 后。例如，**str₁** 输入为 “123456789”，**str₂** 输入为 “abcdefghijk”，则输出的 **str₁** 为 “1a2b3c4d5e6f7g8h9ijk”。
3. 在屏幕上输出新生成的 **str₁**。

提示： 合并时可使用中间数组。

程序的运行效果应类似地如图 808.1 所示，金色部分是从键盘输入的内容。



图 808.1

附件:  P808.c

文件内容:

P808.c

```

#include <stdio.h>
#include <string.h>

#define N 128

void conj(char *string1, char *string2);

int main(void)

```

```

{
    char str1[N * 2], str2[N];

    printf("Please input string1:");
    gets(str1);
    printf("Please input string2:");
    gets(str2);

    /* 本部分代码功能建议：调用函数conj()完成str1和str2的合并 */
    /* User Code Begin(Limit: lines<=1, lineLen<=50, 考生可在本行后添加代码、最多1行、行长<=50字符) */
    <A>
    /* User Code End(考生添加代码结束。注意：空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

    printf("\nstring1:%s\n", str1);

    return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分，如函数conj，行数不限) */
<B>

```

参考答案:

```

<A> conj(str1, str2);
<B> void conj(char *string1, char *string2)
{
    char string3[N * 2];
    int i, j;

    for (i = 0, j = 0; string1[i] != '\0' && string2[i] != '\0'; i++, j += 2)
    {
        string3[j] = string1[i];
        string3[j + 1] = string2[i];
    }

    if (strlen(string1) > strlen(string2))
    {
        for (; string1[i] != '\0'; i++, j++)
        {
            string3[j] = string1[i];
        }
    }
    else
    {
        for (; string2[i] != '\0'; i++, j++)
        {
            string3[j] = string2[i];
        }
    }
    string3[j] = '\0';
    strcpy(string1, string3);
}

```

P825

DIFFICUTLY ★★

从键盘读入一个长整型数，若该数不在 0~2147483647 范围内则要求重输，直到符合要求。然后将该数转换成大写的金额输出。

注：大写的数字为零、壹、贰、叁、肆、伍、陆、柒、捌、玖、拾、佰、仟、万、亿。

可用素材： `printf("请输入一个整数：")`
`printf("\n数据错误，请重新输入：")`
`printf("转换后的结果是：")`

程序的运行效果应类似地如图 825.1、图 825.2、图 825.3、图 825.4 和图 825.5 所示，金色部分是从键盘输入的内容。

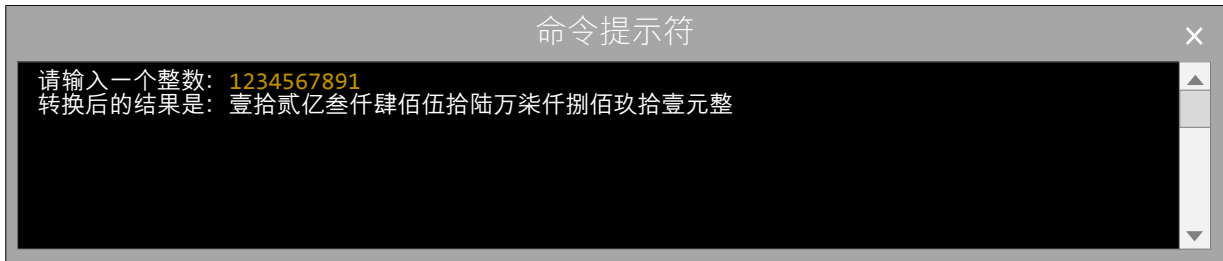


图 825.1 输入“1234567891”时



图 825.2 输入“2147485647”“-10”“10023”时



图 825.3 输入“1000”时



图 825.4 输入“10”时



图 825.5 输入“0”时

参考答案：

```

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

int main(void)
{
    long num;
    int len, i, j;
    char str[10];

    printf("请输入一个整数: ");
    scanf("%ld", &num);

    while (num < 0 || num > 2147483647)
    {
        printf("\n数据错误, 请重新输入: ");
        scanf("%ld", &num);
    }

    itoa(num, str, 10); /* 对于较高版本的编译器, 请使用 _itoa */
    len = strlen(str);

    printf("转换后的结果是: ");
    for (i = 0, j = len; i < len; i++, j--)
    {
        switch (str[i])
        {
            case '0':
                printf("零");
                break;
            case '1':
                printf("壹");
                break;
            case '2':
                printf("贰");
                break;
            case '3':
                printf("叁");
                break;
            case '4':
                printf("肆");
                break;
            case '5':
                printf("伍");
                break;
            case '6':
                printf("陆");
                break;
            case '7':
                printf("柒");
                break;
            case '8':
                printf("捌");
                break;
            case '9':
                printf("玖");
                break;
            default:
                ;
        }
        switch (j)
        {
            case 2:

```

```

        case 6:
        case 10:
            printf("拾");
            break;
        case 3:
        case 7:
            printf("佰");
            break;
        case 4:
        case 8:
            printf("仟");
            break;
        case 5:
            printf("万");
            break;
        case 9:
            printf("亿");
            break;
        default:
            ;
    }
}
printf("元整\n");

return 0;
}

```

P421

DIFFICUTLY ★★

程序 P421.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。完全实现程序功能的程序代码 P421.txt（见文件内容）可供参考。

程序的功能是：

1. 程序运行时先显示“Please input numbers:”，再从键盘上读入一组整数（只考虑 int 型），数与数之间只使用空格或回车作分隔。数可正可负，最多 10000 个，但若读入的数为 -222 时，则表示输入结束且 -222 不算在该组数内。
2. 对这一组数按从小到大的顺序进行排序。
3. 将排序后的这一组数输出到屏幕上，输出格式为每行 6 个数，数与数之间使用逗号“,”分隔，两个逗号之间的宽度（不算逗号）为 6 且使用左对齐格式。注意，行尾没有逗号。

可用素材： `printf("Please input numbers:");`
`printf("\nOutput:\n");`

程序的运行效果应类似地如图 421.1 所示，金色部分是从键盘输入的内容。

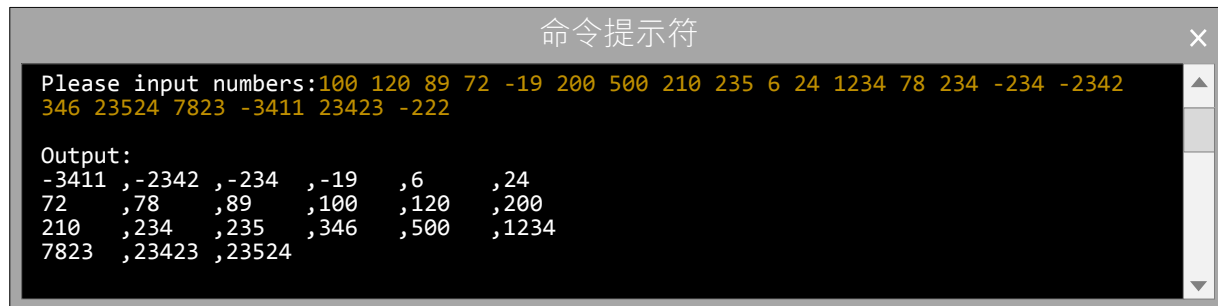
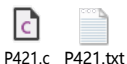


图 421.1

附件:



P421.c P421.txt

文件内容:

P421.c

```
#include <stdio.h>

#define maxNums 10000
#define endFlag -222

/* 本部分代码功能建议: 函数原型声明 */
/* User Code Begin(Limit: lines<=3, lineLen<=50, 考生可在本行后添加代码、最多3行、每行长<=50字符) */
<A>
/* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

int main(void)
{
    int inputData[maxNums] = {0}, dataCount; /* inputData用于保存输入的数据, dataCount记录输入数据的个数 */

    /* 本部分代码功能建议: 调用相应的函数先后完成数据的输入、排序和输出 */
    /* User Code Begin(Limit: lines<=3, lineLen<=40, 考生可在本行后添加代码、最多3行、行长<=40字符) */
    <B>
    /* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

    return 0;
}

/* 本部分代码功能建议: 完成数据的输入 */
/* User Code Begin(Limit: lines<=11, lineLen<=60, 考生可在本行后添加代码、最多11行、每行长<=60字符) */
<C>
/* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */
}

/* 本部分代码功能建议: 对数据进行排序 */
/* User Code Begin(Limit: lines<=11, lineLen<=60, 考生可在本行后添加代码、最多11行、每行长<=60字符) */
<D>
/* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */
}

/* 本部分代码功能建议: 按格式要求输出 */
/* User Code Begin(Limit: lines<=10, lineLen<=60, 考生可在本行后添加代码、最多10行、每行长<=60字符) */
<E>
/* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */
}
/* Program End(程序到此结束, 此后不能添加内容, 否则0分) */
```

P421.txt

```
#include <stdio.h>

#define maxNums 10000
#define endFlag -222
```

```

int main(void)
{
    int i, j, mini, n = 0, x, tmp;
    int data[maxNums] = {0};

    printf("Please input numbers:");
    for (i=0; i<maxNums; i++)
    {
        scanf("%d", &x);
        if (endFlag == x)
        {
            break;
        }
        data[i] = x;
        n++;
    }

    for (i=0; i<n-1; i++)
    {
        mini = i;
        for (j=i; j<n; j++)
        {
            if (data[j] < data[mini])
            {
                mini = j;
            }
        }

        tmp = data[mini];
        data[mini] = data[i];
        data[i] = tmp;
    }

    printf("\nOutput:\n");
    for (i=0; i<n; i++)
    {
        printf("%-6d", data[i]);

        if (((i + 1) % 6) != 0 && i != n - 1)
        {
            printf(",");
        }
        else
        {
            printf("\n");
        }
    }

    return 0;
}

```

参考答案:

```

<A> int Input(int data[]);
    void Sort(int data[], int count);
    void Output(int data[], int count);
<B> dataCount = Input(inputData);
    Sort(inputData, dataCount);
    Output(inputData, dataCount);

```

```

<C> int Input(int data[])
{
    int i, j, tmp;

    printf("Please input numbers:");
    for (i = 0, j = 0; i < maxNums; i++, j++)
    {
        scanf("%d", &tmp);
        if (tmp == endFlag)
        {
            break;
        }
        data[i] = tmp;
    }

    return j;
}

<D> void Sort(int data[], int count)
{
    int i, j, mini, tmp;

    for (i = 0; i < count - 1; i++)
    {
        mini = i;
        for (j = i; j < count; j++)
        {
            if (data[j] < data[mini])
            {
                mini = j;
            }
        }
        tmp = data[mini];
        data[mini] = data[i];
        data[i] = tmp;
    }
}

<E> void Output(int data[], int count)
{
    int i;

    printf("\nOutput:\n");
    for (i = 0; i < count; i++)
    {
        if ((i + 1) % 6 == 0 || i == count - 1)
        {
            printf("%-6d\n", data[i]);
        }
        else
        {
            printf("%-6d,", data[i]);
        }
    }
}

```

P422

DIFFICUTLY ★★

程序 P422.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。完全实现程序功能的程序代码 P422.txt（见文件内容）可供参考。

程序的功能是：

1. 程序运行时先显示“Please input numbers:”，再从键盘上读入一组整数（只考虑 int 型），数与数之间只使用空格或回车作分隔。数可正可负，最多 10000 个，但若读入的数为 -222 时，则表示

输入结束且 -222 不算在该组数内。

2. 对这一组数按从小到大的顺序进行排序。
3. 将排序后的这一组数输出到屏幕上，输出格式为每行 6 个数，数与数之间使用逗号“,”分隔，两个逗号之间的宽度（不算逗号）为 6 且使用左对齐格式。注意，行尾没有逗号。

程序的运行效果应类似地如图 422.1 所示，金色部分是从键盘输入的内容。

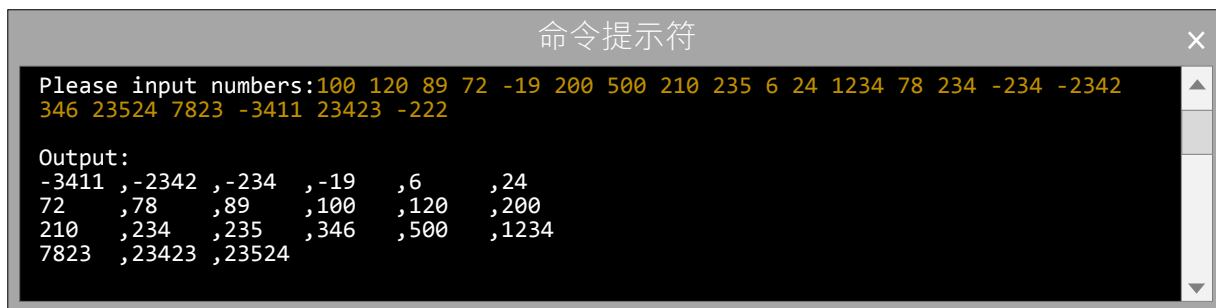


图 422.1

附件:  P422.c  P422.txt

文件内容:

P422.c

```
#include <stdio.h>

#define maxNums 10000
#define endFlag -222

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int i, j, mini;
    int inputData[maxNums] = {0}, dataCount=0; /* inputData用于保存输入的数据，dataCount记录输入数据的个数 */

    printf("Please input numbers:");
    for (i=0; i<maxNums; i++)
    {
        scanf("%d", &inputData[i]);
        if (endFlag == inputData[i])
        {
            break;
        }
        dataCount++;
    }

    for (i=0; i<dataCount-1; i++)
    {
        mini = i;
        for (j=i; j<dataCount; j++)
        {
            if (inputData[j] < inputData[mini])
            {
                mini = j;
            }
        }

        <B> /* userCode(<50字符): 调用函数交换数据的位置 */
    }
}
```

```

printf("\nOutput:\n");
for (i=0; i<dataCount; i++)
{
    printf("%-6d", inputData[i]);

    if (((i + 1) % 6) != 0 && i != dataCount - 1)
    {
        printf(",");
    }
    else
    {
        printf("\n");
    }
}

return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
<C>

```

P422.txt

```

#include <stdio.h>

#define maxNums 10000
#define endFlag -222

int main(void)
{
    int i, j, mini, n=0, tmp;
    int data[maxNums] = {0};

    printf("Please input numbers:");
    for (i=0; i<maxNums; i++)
    {
        scanf("%d", &data[i]);
        if (endFlag == data[i])
        {
            break;
        }
        n++;
    }

    for (i=0; i < n-1; i++)
    {
        mini = i;
        for (j=i; j<n; j++)
        {
            if (data[j] < data[mini])
            {
                mini = j;
            }
        }

        tmp = data[mini];
        data[mini] = data[i];
        data[i] = tmp;
    }
}

```

```

printf("\nOutput:\n");
for (i=0; i<n; i++)
{
    printf("%-6d", data[i]);

    if (((i + 1) % 6) != 0 && i != n - 1)
    {
        printf(",");
    }
    else
    {
        printf("\n");
    }
}

return 0;
}

```

参考答案:

```

<A> void Swap(int *data1, int *data2);
<B> Swap(&inputData[mini], &inputData[i]);
<C> void Swap(int *data1, int *data2)
{
    int tmp;

    tmp = *data1;
    *data1 = *data2;
    *data2 = tmp;
}

```

P423

DIFFICUTLY ★★

程序 P423.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。完全实现程序功能的程序代码 P423.txt（见文件内容）可供参考。

程序的功能是：

1. 程序运行时先显示“Please input numbers:”，再从键盘上读入一组整数（只考虑 int 型），数与数之间只使用空格或回车作分隔。数可正可负，最多 10000 个，但若读入的数为 -222 时，则表示输入结束且 -222 不算在该组数内。
2. 对这一组数按从小到大的顺序进行排序。
3. 将排序后的这一组数输出到屏幕上，输出格式为每行 6 个数，数与数之间使用逗号“,”分隔，两个逗号之间的宽度（不算逗号）为 6 且使用左对齐格式。注意，行尾没有逗号。

程序的运行效果应类似地如图 423.1 所示，金色部分是从键盘输入的内容。

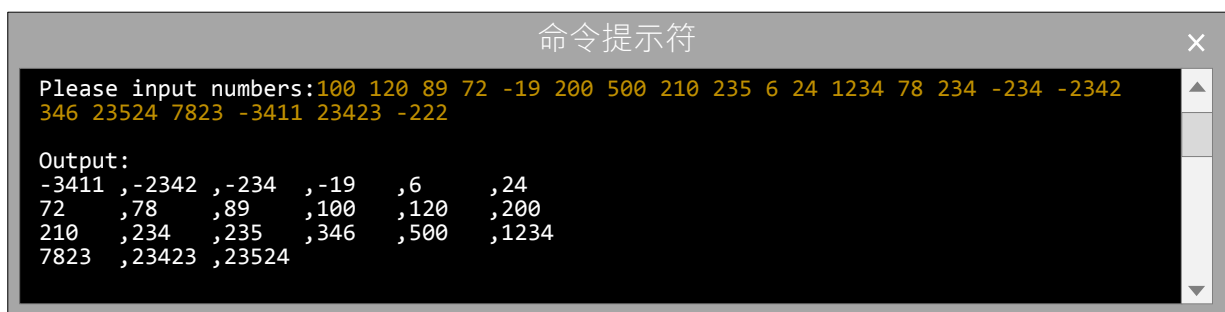
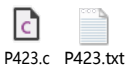


图 423.1

附件:



P423.c P423.txt

文件内容:

P423.c

```
#include <stdio.h>

#define maxNums 10000
#define endFlag -222

/* userCode(<50字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int i, j, mini;
    int inputData[maxNums] = {0}, dataCount=0; /* inputData用于保存输入的数据, dataCount记录输入
数据的个数 */

    printf("Please input numbers:");
    for (i=0; i<maxNums; i++)
    {
        scanf("%d", &inputData[i]);
        if (endFlag == inputData[i])
        {
            break;
        }
        dataCount++;
    }

    for (i=0; i<dataCount-1; i++)
    {
        mini = i;
        for (j=i; j<dataCount; j++)
        {
            if (inputData[j] < inputData[mini])
            {
                mini = j;
            }
        }

        <B> /* userCode(<50字符): 调用函数交换数据的位置 */
    }

    printf("\nOutput:\n");
    for (i=0; i<dataCount; i++)
    {
        printf("%-6d", inputData[i]);

        if ((i + 1) % 6) != 0 && i != dataCount - 1)
        {
            printf(",");
        }
        else
        {
            printf("\n");
        }
    }

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

<C>

P423.txt

```
#include <stdio.h>

#define maxNums 10000
#define endFlag -222

int main(void)
{
    int i, j, mini, n=0, tmp;
    int data[maxNums] = {0};

    printf("Please input numbers:");
    for (i=0; i<maxNums; i++)
    {
        scanf("%d", &data[i]);
        if (endFlag == data[i])
        {
            break;
        }
        n++;
    }

    for (i=0; i <n-1; i++)
    {
        mini = i;
        for (j=i; j<n; j++)
        {
            if (data[j] < data[mini])
            {
                mini = j;
            }
        }

        tmp = data[mini];
        data[mini] = data[i];
        data[i] = tmp;
    }

    printf("\nOutput:\n");
    for (i=0; i<n; i++)
    {
        printf("%-6d", data[i]);

        if (((i + 1) % 6) != 0 && i != n - 1)
        {
            printf(",");
        }
        else
        {
            printf("\n");
        }
    }

    return 0;
}
```

参考答案:

```
<A> int GetMin(int data[], int count, int i);
<B> mini = GetMin(inputData, dataCount, i);
<C> int GetMin(int data[], int count, int i)
{
    int min = i, minval = data[i], j;

    for (j = i; j < count; j++)
    {
        if (data[j] < minval)
        {
            min = j;
            minval = data[j];
        }
    }

    return min;
}
```

P707

DIFFICUTLY ★★

下面是一个五阶的螺旋方阵。编程输出此形式的 n ($n \leq 15$) 阶的方阵（顺时针方向旋进）， n 由键盘输入。输出时要求每个数据宽度为 5、右对齐。

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 16 & 17 & 18 & 19 & 6 \\ 15 & 24 & 25 & 20 & 7 \\ 14 & 23 & 22 & 21 & 8 \\ 13 & 12 & 11 & 10 & 9 \end{pmatrix}$$

可用素材: `printf("Enter n(n<=15):\n")`

程序的运行效果应类似地如图 707.1 所示，金色部分是从键盘输入的内容。

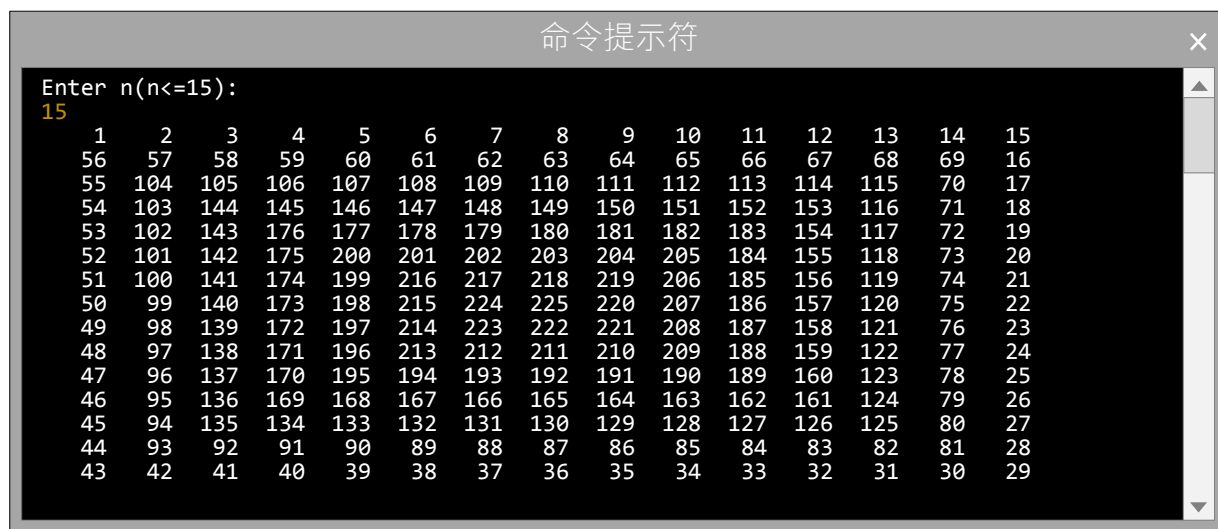


图 707.1

参考答案:

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

```
int main(void)
```

```
{
    int arr[15][15], m0, n0, i, j, k = 1;
```

```

do
{
    printf("Enter n(n<=15):\n");
    scanf("%d", &n0);
} while (n0 < 0 || n0 > 15);

if (n0 % 2 == 0)
{
    m0 = n0 / 2;
}
else
{
    m0 = n0 + 1;
}

for (i = 0; i < m0; i++)
{
    for (j = i; j < n0 - i; j++)
    {
        arr[i][j] = k;
        k++;
    }
    for (j = i + 1; j < n0 - i; j++)
    {
        arr[j][n0 - i - 1] = k;
        k++;
    }
    for (j = n0 - i - 2; j >= i; j--)
    {
        arr[n0 - i - 1][j] = k;
        k++;
    }
    for (j = n0 - i - 2; j >= i + 1; j--)
    {
        arr[j][i] = k;
        k++;
    }
}

for (i = 0; i < n0; i++)
{
    for (j = 0; j < n0; j++)
    {
        printf("%5d", arr[i][j]);
    }
    putchar('\n');
}

return 0;
}

```

P726

DIFFICUTLY ★★

从键盘上读入一行字符，在屏幕上输出该行字符的长度及内容（先输出长度，后输出内容）。

注意：

1. 以回车表示行结束且回车不计入输入内容。若读入过程中发生错误或遇到文件结束，则也表示行输入结束。
2. 若用户输入时输入了很多字符，则仅读入前 100 个字符。

3. 不能使用库函数 gets()、fgets()、strlen() 或使用同名的变量、函数、单词。

可用素材: `printf("input a string: ")`
`printf("\nThe string lenth is: ...`
`printf("\nThe string is: ...`

程序的运行效果应类似地如图 726.1 所示，金色部分是从键盘输入的内容。



图 726.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int ch, i, len = 0;
    char str[101];

    printf("input a string: ");
    for (i = 0; i < 101; i++)
    {
        ch = getchar();
        if (ch == EOF || ch == '\n' || i == 100)
        {
            break;
        }
        else
        {
            str[i] = ch;
            len++;
        }
    }
    str[i] = '\0';
    printf("\nThe string lenth is: %d", len);
    printf("\nThe string is: %s\n", str);

    return 0;
}
```

P741

DIFFICUTLY ★★

输入两个整数 m 和 n ，输出大于等于 m ($m > 5$) 的 n 个素数，输出的各素数间以空格相隔。

注：素数 (Prime Number)，亦称质数，指在一个大于 1 的自然数中，除了 1 和此整数自身外，没法被其他自然数整除的数。

可用素材: `printf("Input the m, n: ")`
`printf("\nThe result:\n")`

程序的运行效果应类似地如图 741.1 所示，金色部分是从键盘输入的内容。



图 741.1

参考答案:

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

```
int main(void)
{
    int m0, n0, i, flag = 0, count = 0, j;

    printf("Input the m, n: ");
    scanf("%d,%d", &m0, &n0);

    printf("\nThe result:\n");
    for (j = m0; ; j++)
    {
        flag = 0;

        for (i = 2; i < j; i++)
        {
            if (j % i == 0)
            {
                flag = 1;
                break;
            }
        }
        if (flag == 0)
        {
            printf("%d ", i);
            count++;
        }
        if (count == n0)
        {
            break;
        }
    }
    putchar('\n');

    return 0;
}
```

P746

DIFFICUTLY ★★

从键盘读入 10 个整数，对其按由小到大的顺序进行排序，然后输出。

可用素材: `printf("please input 10 integer numbers: ")`
`printf("\nthe array before sorted: ")`
`printf("\nthe array after sorted: ")`

程序的运行效果应类似地如图 746.1 所示，金色部分是从键盘输入的内容。

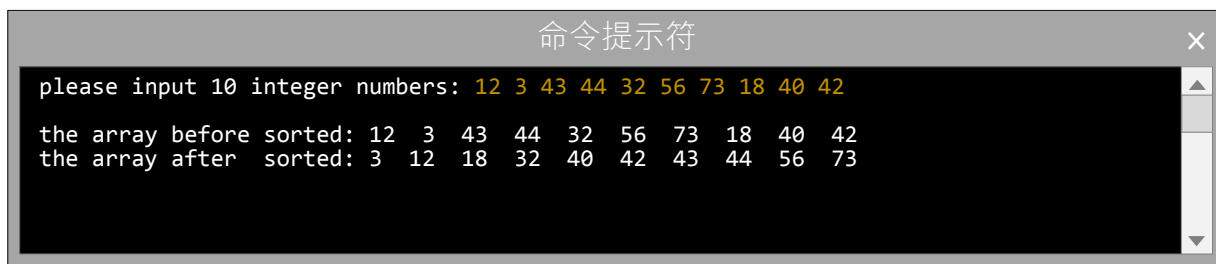


图 746.1

参考答案:

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

```

int main(void)
{
    int arr[10], i, j, tmp;

    printf("please input 10 integer numbers: ");
    for (i = 0; i < 10; i++)
    {
        scanf("%d", &arr[i]);
    }

    printf("\nthe array before sorted: ");
    for (i = 0; i < 10; i++)
    {
        printf("%d ", arr[i]);
    }
    printf("\nthe array after sorted: ");
    for (j = 1; j < 10; j++)
    {
        for (i = 0; i < 10 - j; i++)
        {
            if (arr[i] > arr[i + 1])
            {
                tmp = arr[i];
                arr[i] = arr[i + 1];
                arr[i + 1] = tmp;
            }
        }
    }

    for (i = 0; i < 10; i++)
    {
        printf("%d ", arr[i]);
    }
    putchar('\n');

    return 0;
}

```

P751

DIFFICUTLY ★★

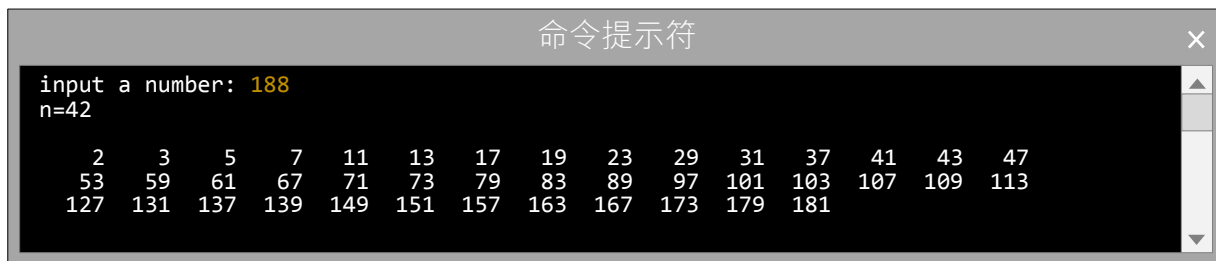
输入整数 m ，将所有大于 1 小于整数 m 的素数存入所指定的数组中（数组最多只存放 100 个素数，超过则提示“overflow”），输出素数的个数 n 及各素数——素数的输出格式为每个素数 5 列宽、右对齐、每行显示 15 个。若输入的 $m \leq 0$ ，则提示“error”，程序终止。

注：素数（Prime Number），亦称质数，指在一个大于 1 的自然数中，除了 1 和此整数自身外，没法

被其他自然数整除的数。

可用素材: `printf("input a number: ")`
`printf("error")`
`printf("overflow")`
`printf("n=...\n"...`

程序的运行效果应类似地如图 751.1、图 751.2 和图 751.3 所示，金色部分是从键盘输入的内容。



```
命令提示符
input a number: 188
n=42
  2   3   5   7  11  13  17  19  23  29  31  37  41  43  47
53  59  61  67  71  73  79  83  89  97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173 179 181
```

图 751.1



```
命令提示符
input a number: 0
error
```

图 751.2



```
命令提示符
input a number: 568
overflow
```

图 751.3

参考答案:

```
#include <stdio.h>

int main(void)
{
    int num, arr[100], i, j, k, count = 0, flag;

    printf("input a number: ");
    scanf("%d", &num);

    if (num <= 0)
    {
        printf("error");
    }
    else
    {
        for (i = 2; i < num; i++, flag = 0)
        {
            for (j = 2; j < i; j++)
            {
                if (i != 2 && i % j == 0)
                {
                    flag = 1;
                }
            }
        }
    }
}
```

```

        }
    }
    if (flag == 0)
    {
        count++;
    }
}
if (num != 2)
{
    count++;
}

if (count > 100)
{
    printf("overflow");
}
else
{
    for (i = 2, k = 0; i < num; i++, flag = 0)
    {
        for (j = 2; j < i; j++)
        {
            if (i != 2 && i % j == 0)
            {
                flag = 1;
            }
        }
        if (flag == 0)
        {
            arr[k] = i;
            k++;
        }
    }
    printf("n=%d\n", count);
    for (i = 0; i < count; i++)
    {
        if ((i + 1) % 15 == 0)
        {
            printf("%5d\n", arr[i]);
        }
        else
        {
            printf("%5d", arr[i]);
        }
    }
}
}
putchar('\n');

return 0;
}

```

P775

DIFFICUTLY ★★

从键盘上输入 2 个字符串（约定：每个字符串中字符数 ≤ 80 字节，字符串中可以有空格），找出这两个串的最大公约数字符串（即拥有的相同的最长长度的子串，若有多个，则显示第 1 个），如字符串 1 为“abcdefg”、字符串 2 为“acdefm4365”，则其最大公约数串是“cdef”。

可用素材： printf("the First string is: ")

```
printf("the Second string is: ")
printf("\nResult = (%s)\n"...
```

程序的运行效果应类似地如图 775.1、图 775.2、图 775.3 所示，金色部分是从键盘输入的内容。

图 775.1

图 775.2

图 775.3

参考答案:

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

int main(void)
{
    char str1[81], str2[81], result[81];
    int start = -1, end = -1, max = 0, i, j, dp[100][100], s0[100][100], flag = 0;

    printf("the First string is: ");
    gets(str1);
    printf("the Second string is: ");
    gets(str2);

    memset(dp, 0, sizeof(dp));
    for (i = 0; str1[i] != '\0'; i++)
    {
        s0[i][0] = i + 1;
    }
    for (i = 0; str1[i] != '\0'; i++)
    {
        for (j = 0; str2[j] != '\0'; j++)
        {
            if (str1[i] == str2[j])
            {
                dp[i + 1][j + 1] = dp[i][j] + 1;
                s0[i + 1][j + 1] = s0[i][j];
            }
        }
    }
}
```

```

        flag = 1;
    }
    else
    {
        dp[i + 1][j + 1] = 0;
        s0[i + 1][j + 1] = i + 2;
    }
    if (dp[i + 1][j + 1] > max)
    {
        max = dp[i + 1][j + 1];
        end = i;
        start = s0[i + 1][j + 1] - 1;
    }
}
}
if (flag == 1)
{
    for (i = start, j = 0; i <= end; i++)
    {
        result[j] = str1[i];
        j++;
    }
    result[j] = '\0';
    printf("\nResult = (%s)\n", result);
}
else
{
    printf("\nResult = ()\n");
}

return 0;
}

```

P777

DIFFICUTLY ★★

方阵的主对角线之上称为“上三角”，设计一个用于填充 n (n 从键盘读入，约定其取值范围为 3~20) 阶方阵的上三角区域的程序。填充的规则是：使用 1, 2, 3, ... 的自然数列，从左上角开始，按照顺时针方向螺旋填充。输出时要求每个数据宽度为 4、右对齐。

可用素材：

```

printf("Please input n: ")
printf("\nResult is:\n")
printf("%4d"...
printf("\n")

```

程序的运行效果应类似地如图 777.1 所示，金色部分是从键盘输入的内容。



图 777.1

参考答案:

```
#include <stdio.h>

int main(void)
{
    int arr[20][20], n0, i, j, k = 1;

    do
    {
        printf("Please input n: ");
        scanf("%d", &n0);
    } while (n0 < 3 || n0 > 20);

    for (i = 0; i < (n0 - 1) / 2; i++)
    {
        for (j = i; j <= n0 - (2 * i + 1); j++)
        {
            arr[i][j] = k;
            k++;
        }
        for (j = i + 1; j <= n0 - (2 * i + 1); j++)
        {
            arr[j][n0 - j - i - 1] = k;
            k++;
        }
        for (j = n0 - (2 * i + 1) - 1; j >= i + 1; j--)
        {
            arr[j][i] = k;
            k++;
        }
    }

    printf("\nResult is:\n");
    for (i = 0; i < n0; i++)
    {
        for (j = 0; j <= n0 - i - 1; j++)
        {
            printf("%4d", arr[i][j]);
        }
        putchar('\n');
    }

    return 0;
}
```

P804

DIFFICUTLY ★★

程序 P804.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：先从键盘读入的 8 个学生的 6 门课成绩存放在二维数组 **student** 中（每一行存储一个学生的数据，第 0 列为学号，第 1~6 列依次为 6 门课的成绩），再从键盘读入一个学号，在 **student** 查找该学生是否存在，若存在，则输出其平均成绩，若不存在，则显示“not Find!”。

程序的运行效果应类似地如图 804.1 所示，金色部分是从键盘输入的内容。




```
命令提示符
Input the 8 student's number and score:
1: 20011 100 80 90 78 85 92
2: 20012 90 83 88 76 80 93
3: 20015 85 50 73 75 83 90
4: 20013 89 80 78 83 75 52
5: 20021 60 82 85 98 75 76
6: 20031 70 70 63 68 93 66
7: 20035 78 60 72 82 88 57
8: 20026 92 85 99 95 65 81

Input a student's number to compute: 20035

The No.20035 student's average is 72.83
```

图 804.1

附件:  P804.c

文件内容:

P804.c

```
#include<stdio.h>

#define NOT_FIND -1
#define TOTAL_STU 8

/* 函数aver的功能为: 求学号为stuNo的学生的6门课课程之平均成绩, 并通过函数值返回
   若未找到学生stuNo, 则返回NOT_FIND */
float aver(int (*pStu)[7], int stuNo);

int main(void)
{
    int student[TOTAL_STU][7]; /* the first column save student's number */
    float averScore;
    int i, j, stuNumber;

    printf("Input the %d student's number and score: \n", TOTAL_STU);
    /* 本部分代码功能建议: 从键盘读入的8个学生的6门课成绩存放在二维数组student中 */
    /* User Code Begin(Limit: lines<=6, lineLen<=50, 考生可在本行后添加代码、最多6行、行长<=50字符) */
    <A>
    /* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

    printf("\nInput a student's number to compute: ");
    scanf("%d", &stuNumber);

    /* 本部分代码功能建议: 调用函数aver()求平均值 */
    /* User Code Begin(Limit: lines<=1, lineLen<=50, 考生可在本行后添加代码、最多1行、行长<=50字符) */
    <B>
    /* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */
    if (NOT_FIND == averScore)
    {
        printf("\nstudent of No.%d not Find!\n", stuNumber);
    }
    else
    {
        printf("\nThe No.%d student's average is %.2f\n", stuNumber, averScore);
    }

    return 0;
}
```



```
/* User Code Begin(考生在此后根据设计需要完成程序的其它部分，如函数aver，行数不限) */
<C>
```


参考答案:

```
<A> for (i = 0; i < TOTAL_STU; i++)
{
    printf("%d: ", i + 1);
    for (j = 0; j < 7; j++)
    {
        scanf("%d", &student[i][j]);
    }
}
<B> averScore = aver(student, stuNumber);
<C> float aver(int (*pStu)[7], int stuNo)
{
    int i, j, sum = 0;
    float avg;

    for (i = 0; i < TOTAL_STU; i++)
    {
        if (pStu[i][0] == stuNo)
        {
            for (j = 1; j < 7; j++)
            {
                sum += pStu[i][j];
            }
            avg = sum / (float)6;
            return avg;
        }
    }
    return NOT_FIND;
}
```

P761

DIFFICUTLY ★★

程序 P761.c 已编写部分代码（见文件内容），请根据程序中的要求编写函数 Move() 的代码（在指定的位置添加代码或将  换成代码）。

函数 void move(int array[], int n, int m) 的功能为：在数组 **array** 中有 n 个整数，使其前面各数顺序向后移 m ($0 \leq m \leq n$) 个位置，最后 m 个数变成最前面的 m 个数，编写该函数。

程序的运行效果应类似地如图 761.1 所示，金色部分是从键盘输入的内容。



图 761.1

附件:  P761.c

文件内容:

P761.c

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

/* User Code Begin(考生可在本行后添加代码, 例如全局变量的定义、函数原型声明等, 行数不限) */
<A>
/* User Code End(考生添加代码结束) */

int main(void)
{
    int *number, n, m, i;

    printf("the total numbers is: ");
    scanf("%d", &n);
    printf("back m: ");
    scanf("%d", &m);

    number = (int *)malloc(n * sizeof(int));
    if (NULL == number)
    {
        puts("memory allocation failure!");
        exit(1);
    }

    printf("input %d integers: ", n);
    for (i=0; i<n; i++)
    {
        scanf("%d", &number[i]);
    }

    move(number, n, m);

    printf("\n    after move %d: ", m);
    for (i=0; i<n; i++)
    {
        printf("%d ", number[i]);
    }
    printf("\n");

    free(number);
    return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分, 如函数move, 行数不限) */
<B>
```

参考答案:

```
<A> void move(int *num, int n, int m);
<B> void move(int *num, int n, int m)
{
    int i, j, *pm, count = 0;

    pm = (int *)malloc(m * sizeof(int));
    for (i = n - m, j = 0; i <= n - 1; i++, j++)
    {
        *(pm + j) = *(num + i);
    }
}
```

```

    }
    for (i = n - m - 1, j = n - 1; i >= 0; i--, j--)
    {
        *(num + j) = *(num + i);
    }
    for (i = 0; i < m; i++)
    {
        *(num + i) = *(pm + i);
    }
    free(pm);
}

```

P762

DIFFICUTLY ★★

程序 P762.c 已编写部分代码（见文件内容），请根据程序中的要求编写函数 FindMin() 的代码（在指定的位置添加代码或将 `<A>` 换成代码）。

函数的功能为：从键盘读入 10 个数存入数组中，找出并显示最小元素及其在数组中的位置。

程序的运行效果应类似地如图 762.1 所示，金色部分是从键盘输入的内容。

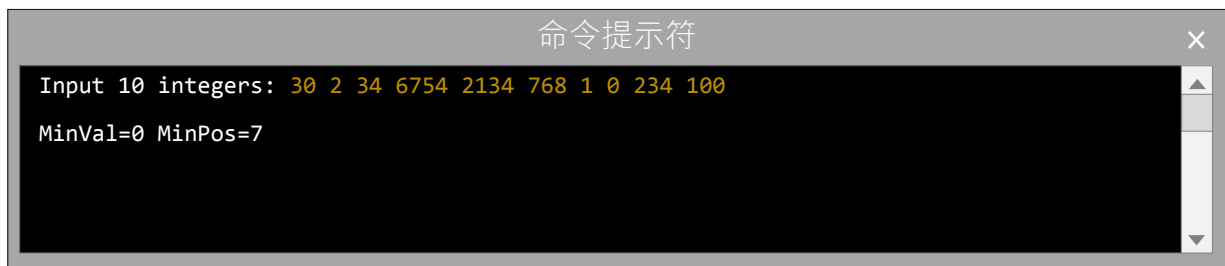



图 762.1

附件:  P762.c

文件内容:

P762.c

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

```
/* userCode(<80字符>): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int num[10], i, MinVal, MinPos;
```

```
    printf("Input 10 integers: ");
```

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

```
    {
```

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

```
    }
```

```
    MinVal = FindMin(num, 10, &MinPos);
```

```
    printf("\nMinVal=%d MinPos=%d\n", MinVal, MinPos);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<B>
```

参考答案:


```
<A> int FindMin(int num[], int n, int *pMinPos);
<B> int FindMin(int num[], int n, int *pMinPos)
{
    int i, min = num[0];

    *pMinPos = 0;
    for (i = 1; i < n; i++)
    {
        if (num[i] < min)
        {
            min = num[i];
            *pMinPos = i;
        }
    }

    return min;
}
```

P763

DIFFICUTLY ★★

程序 P763.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：有五个学生，每个学生的数据包括学号、姓名（最长 19 字节）、三门课的成绩，从键盘输入五个学生的数据，并计算每个学生的平均成绩，最后显示最高平均分的学生的信息（包括学号，姓名，三门课的成绩，平均分数）。

注：要求用结构体编程，变量数据类型的选择应适当，在保证满足设计要求精度的情况下，养成不浪费内存空间和计算时间的好习惯。

可用素材： `printf("Please input students info:Num Name score1 score2 score3\n")`

程序的运行效果应类似地如图 763.1 所示，金色部分是从键盘输入的内容。

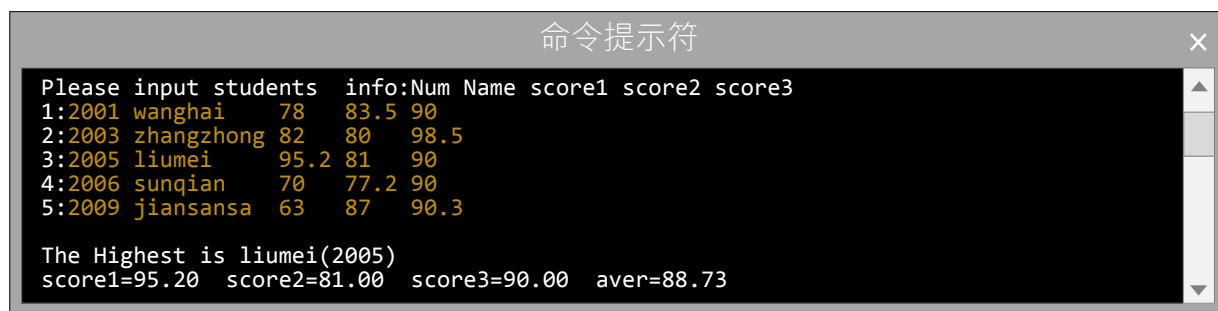


图 763.1

附件：
P763.c

文件内容：

P763.c

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

```
/* User Code Begin(考生可在本行后添加代码，例如全局变量的定义、函数原型声明等，行数不限) */
```

```
<A>
```

```

/* User Code End(考生添加代码结束) */

int main(void)
{
    int high; /* high记录平均分最高的学生的序号，具体使用参考后面的代码 */

    /* User Code Begin(考生可在本行后添加代码，行数不限) */
    <B>
    /* User Code End(考生添加代码结束) */

    printf("\nThe Highest is %s(%d)\n", myClass[high].name, myClass[high].num,
           myClass[high].score1, myClass[high].score2, myClass[high].score3, myClass[high].aver);

    return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分，行数不限) */
<C>
/* User Code End(考生添加代码结束。注意：空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */
}
/* Program End(程序到此结束，此后不能添加内容，否则0分) */

```

参考答案：

```

<A> struct student
{
    int num;
    char name[20];
    float score1, score2, score3, aver;
};

int GetMax(struct student myClass[], int n);
<B> struct student myClass[5];
int i;

printf("Please input students info:Num Name score1 score2 score3\n");
for (i = 0; i < 5; i++)
{
    printf("%d:", i + 1);
    scanf("%d%s%f%f%f", &myClass[i].num, myClass[i].name,
           &myClass[i].score1, &myClass[i].score2, &myClass[i].score3);
    myClass[i].aver = (myClass[i].score1 + myClass[i].score2 + myClass[i].score3) / 3;
}
high = GetMax(myClass, 5);
<C> int GetMax(struct student myClass[], int n)
{
    int i, high = 0;
    float max = myClass[0].aver;

    for (i = 1; i < n; i++)
    {
        if (myClass[i].aver > max)
        {
            max = myClass[i].aver;
            high = i;
        }
    }
}

```

```
        return high;
    }
```

P780

DIFFICUTLY ★★

程序 P780.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。


程序的功能是：从键盘上读入一行字符，在屏幕上输出该行字符的长度及内容（先输出长度，后输出内容），具体要求有：

1. 以回车表示行结束且回车不计入输入内容。若读入过程中发生错误或遇到文件结束，则也表示行输入结束。
2. 若用户输入时输入了很多字符，则仅读入前 100 个字符。
3. 不能使用库函数 `gets()`、`fgets()`、`strlen()` 或使用同名的变量、函数、单词。

程序的运行效果应类似地如图 780.1 所示，金色部分是从键盘输入的内容。



图 780.1

附件:  P780.c

文件内容:

P780.c

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

```
/* userCode(<50字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
```

```
{
```

```
    int Len;
```

```
    char String[101] = "????????????????????????????????????????????????????????????";
```

```
    printf("input a string: ");
```

```
    <B> /* userCode(<50字符): 调用函数实现输入并统计输入字符个数 */
```

```
    printf("\nThe string lenth is: %d\n", Len);
```

```
    printf("The string is: %s\n", String);
```

```
    return 0;
```

```
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
```

```
<C>
```

参考答案:

```

<A> int GetLen(char str[]);
<B> Len = GetLen(String);
<C> int GetLen(char str[])
{
    int ch, i, len = 0;

    for (i = 0; i < 101; i++)
    {
        ch = getchar();

        if (ch == EOF || ch == '\n' || i == 100)
        {
            break;
        }
        else
        {
            str[i] = ch;
            len++;
        }
    }
    str[i] = '\0';

    return len;
}

```

P782

DIFFICUTLY ★★

程序 P782.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：有五个学生，每个学生的数据包括学号、姓名（最长 19 字节）、四门课的成绩，从键盘输入五个学生的数据，并计算每个学生的平均成绩，最后显示最高平均分的学生的信息（包括学号，姓名，四门课的成绩，平均分数）。

注：要求用结构体编程，变量数据类型的选择应适当，在保证满足设计要求精度的情况下，养成不浪费内存空间和计算时间的好习惯。

可用素材： `printf("Please input students info:Num Name score1 score2 score3 score4\n")`

程序的运行效果应类似地如图 782.1 所示，金色部分是从键盘输入的内容。

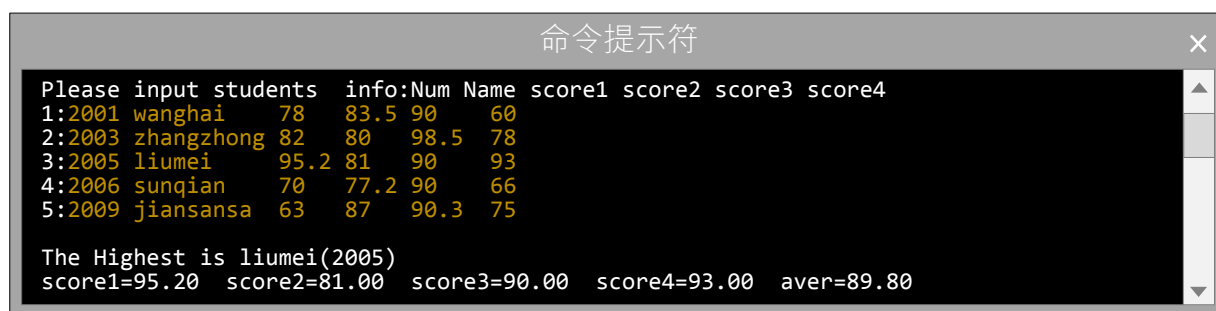


图 782.1

附件： P782.c

文件内容：

P782.c

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

```

/* User Code Begin(考生可在本行后添加代码, 例如全局变量的定义、函数原型声明等, 行数不限) */
<A>
/* User Code End(考生添加代码结束) */

int main(void)
{
    int high; /* high记录平均分最高的学生的序号, 具体使用参考后面的代码 */

    /* User Code Begin(考生可在本行后添加代码, 行数不限) */
    <B>
    /* User Code End(考生添加代码结束) */

    printf("\nThe Highest is %s(%d)\nscore1=%.2f score2=%.2f score3=%.2f score4=%.2f aver=%.2f\n",
           myClass[high].name, myClass[high].num,
           myClass[high].score1, myClass[high].score2, myClass[high].score3, myClass[high].score4,
           myClass[high].aver);

    return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分, 行数不限) */
<C>

```

参考答案:

```

<A> struct student
{
    int num;
    char name[20];
    float score1, score2, score3, score4, aver;
};

int GetMax(struct student myClass[], int n);
<B> struct student myClass[5];
int i;

printf("Please input students info:Num Name score1 score2 score3 score4\n");
for (i = 0; i < 5; i++)
{
    printf("%d:", i + 1);
    scanf("%d%s%f%f%f", &myClass[i].num, myClass[i].name,
           &myClass[i].score1, &myClass[i].score2, &myClass[i].score3, &myClass[i].score4);
    myClass[i].aver = (myClass[i].score1 + myClass[i].score2 + myClass[i].score3 +
                       myClass[i].score4) / 4;
}
high = GetMax(myClass, 5);
<C> int GetMax(struct student myClass[], int n)
{
    int i, high = 0;
    float max = myClass[0].aver;

    for (i = 1; i < n; i++)
    {
        if (myClass[i].aver > max)
        {
            max = myClass[i].aver;
            high = i;
        }
    }
}

```



```

    }
}

return high;
}

```

P783

DIFFICUTLY ★★

程序 P783.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：某班有 4 个学生，三门课，通过 main() 函数读入学生的学号（5 位数字）、姓（最长 7 个字符）和三门课的成绩，然后通过调用用户自定义函数 DispScore() 实现这些信息的输出，调用用户自定义函数 FindNoPass() 实现找出有两门以上不及格的学生、输出其学号和不及格课程的成绩。

注：要求用结构体编程。

可用素材：

```

printf("\nStudent Info and Score:\n")
printf("%6s %8s"...
printf("%5d"...
printf("\nTwo Course No Pass Students:\n")

```

程序的运行效果应类似地如图 783.1 所示，金色部分是从键盘输入的内容。

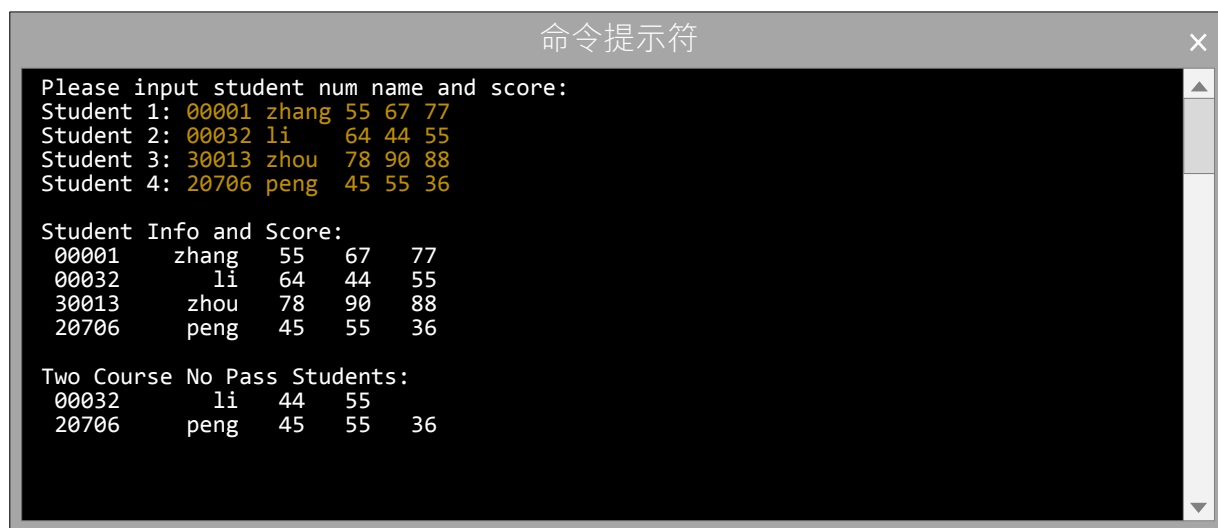



图 783.1

附件:  P783.c

文件内容:

P783.c

```

#include<stdio.h>

#define SNUM 4 /* student number */
#define CNUM 3 /* course number */

/* User Code Begin(考生可在本行后添加代码，例如结构体的定义、函数原型声明等，行数不限) */
<A>

/* User Code End(考生添加代码结束) */

int main(void)

```

```

{
    int i, j;
    struct student stu[SNUM];

    printf("Please input student num name and score:\n");
    for (i=0; i<SNUM; i++)
    {
        printf("Student %d: ", i+1);
        scanf("%s %s", stu[i].num, stu[i].name);
        for (j=0; j<CNUM; j++)
        {
            scanf("%d", &stu[i].score[j]);
        }
    }

    DispScore(stu);
    FindNoPass(stu);

    return 0;
}

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */

```


参考答案:

```

<A> struct student
{
    char num[6], name[8];
    int score[CNUM];
};

void DispScore(struct student stu[]);
void FindNoPass(struct student stu[]);
<B> void DispScore(struct student stu[])
{
    int i, j;

    printf("\nStudent Info and Score:\n");
    for (i = 0; i < SNUM; i++)
    {
        printf("%6s%8s", stu[i].num, stu[i].name);
        for (j = 0; j < CNUM; j++)
        {
            printf("%5d", stu[i].score[j]);
        }
        putchar('\n');
    }

    void FindNoPass(struct student stu[])
    {
        int i, j, flag;

        printf("\nTwo Course No Pass Students:\n");
        for (i = 0; i < SNUM; i++)
        {
            flag = 0;
            if (stu[i].score[0] < 60 && stu[i].score[1] < 60 ||
                stu[i].score[1] < 60 && stu[i].score[2] < 60 ||
                stu[i].score[0] < 60 && stu[i].score[2] < 60)

```

```

    {
        flag = 1;
    }
    if (flag == 1)
    {
        printf("%6s %8s", stu[i].num, stu[i].name);
        for (j = 0; j < CNUM; j++)
        {
            if (stu[i].score[j] < 60)
            {
                printf("%5d", stu[i].score[j]);
            }
        }
        putchar('\n');
    }
}
}
}

```

P784

DIFFICUTLY ★★

程序 P784.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：将 3 位学生 4 门课成绩读入并存储在二维数组 **score** 中，然后输出第 n （约定 $n \geq 2$ ）个学生的成绩。

注：要求用户编程部分对数组 **score** 及其元素的访问必须使用指针实现，即自定义函数头和函数体中不得出现数组下标形式的表示法。

可用素材： `printf("\nthe score of No %d are:"...`
`printf("%6.1f"...`

程序的运行效果应类似地如图 784.1 所示，金色部分是从键盘输入的内容。

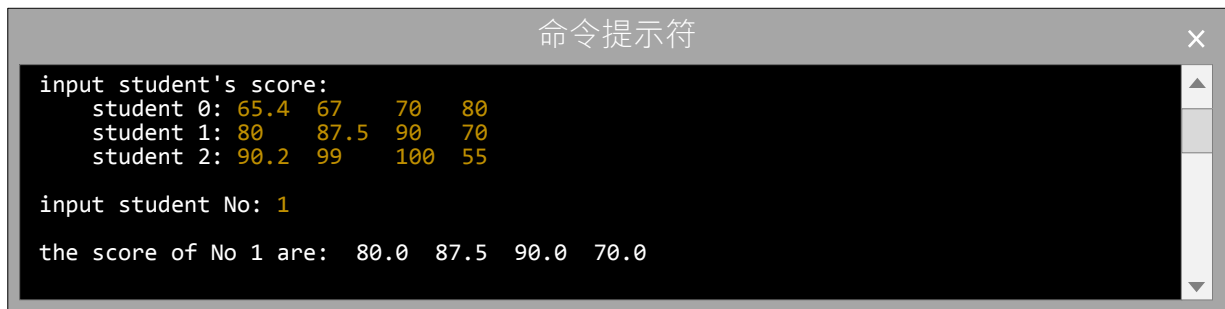


图 784.1

附件: P784.c

文件内容:

P784.c

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

```
/* userCode(<80字符): 自定义函数之原型声明 */
<A>_____
```

```
int main(void)
{
    int n, i;
    float score[3][4];

```

```

printf("input student's score:\n");
for (i=0; i<=2; i++)
{
    printf("    student %d: ", i);
    scanf("%f %f %f %f", &score[i][0], &score[i][1], &score[i][2], &score[i][3]);
}

printf("\ninput student No: ");
scanf("%d", &n);
search(score, n);

return 0;
}

```

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */

参考答案:

```


<A> void search(float (*score)[4], int n);
<B> void search(float (*score)[4], int n)
{
    int i;

    printf("\nthe score of No %d are:", n);
    for (i = 0; i < 4; i++)
    {
        printf("%6.1f", (*(score + n) + i));
    }
    putchar('\n');
}

```

P785

DIFFICUTLY ★★

程序 P785.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：将读入的字符串 s_1 、 s_2 分别调用自定义函数 `myswap()` 反转，然后调用自定义函数 `merge()` 将 s_1 、 s_2 按排列的顺序交叉合并到 s_3 中， s_1 或 s_2 中过长的剩余字符接在 s_3 的尾部。

注：程序中不能使用库函数 `strrev()` 或使用同名的变量、函数、单词。

程序的运行效果应类似地如图 785.1 所示，金色部分是从键盘输入的内容。



图 785.1

附件:  P785.c

文件内容:

P785.c

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

/* myswap完成字符串str内容的反转, 返回字符串str的地址 */
char *myswap(char *str);
/* merge完成字符串strA, strB顺序交叉合并至strC, 返回字符串strC的地址 */
char *merge(char *strA, char *strB, char *strC);

int main(void)
{
    char s1[100], s2[100], s3[200];

    printf("Enter string s1: ");
    gets(s1);
    printf("Enter string s2: ");
    gets(s2);

    printf("\nstring s1 reversed: %s", myswap(s1));
    printf("\nstring s2 reversed: %s", myswap(s2));
    printf("\nstring s1,s2 merged: %s\n", merge(s1, s2, s3));

    return 0;
}

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */
<A>
```

参考答案:

```
<A> char *myswap(char *str)
{
    int i, j, len;
    char tmp[100];

    len = strlen(str);
    for (i = 0, j = len - 1; i < len; i++, j--)
    {
        tmp[j] = *(str + i);
    }
    tmp[len] = '\0';
    strcpy(str, tmp);

    return str;
}

char *merge(char *strA, char *strB, char *strC)
{
    int i, j;

    for (i = 0, j = 0; strA[i] != '\0' && strB[i] != '\0'; i++, j += 2)
    {
        strC[j] = strA[i];
        strC[j + 1] = strB[i];
    }

    if (strlen(strA) > strlen(strB))
    {
        for (; strA[i] != '\0'; i++, j++)
        {
            strC[j] = strA[i];
        }
    }
}
```

```

    }
}
else
{
    for (; strB[i] != '\0'; i++, j++)
    {
        strC[j] = strB[i];
    }
}
strC[j] = '\0';

return strC;
}

```

P786

DIFFICUTLY ★★

程序 P786.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：调用自定义函数 creat() 读入第一部分学生信息建立链表 A、调用自定义函数 creat() 读入第二部分学生信息建立链表 B，每个链表中的节点包括学号、成绩；然后调用自定义函数 merge() 将两个链表以节点的学号升序合并为一个链表 C。三个链表的内容均调用自定义函数 print() 输出在屏幕上。

可用素材： printf("学生 %d: ")...

程序的运行效果应类似地如图 786.1 所示，金色部分是从键盘输入的内容。

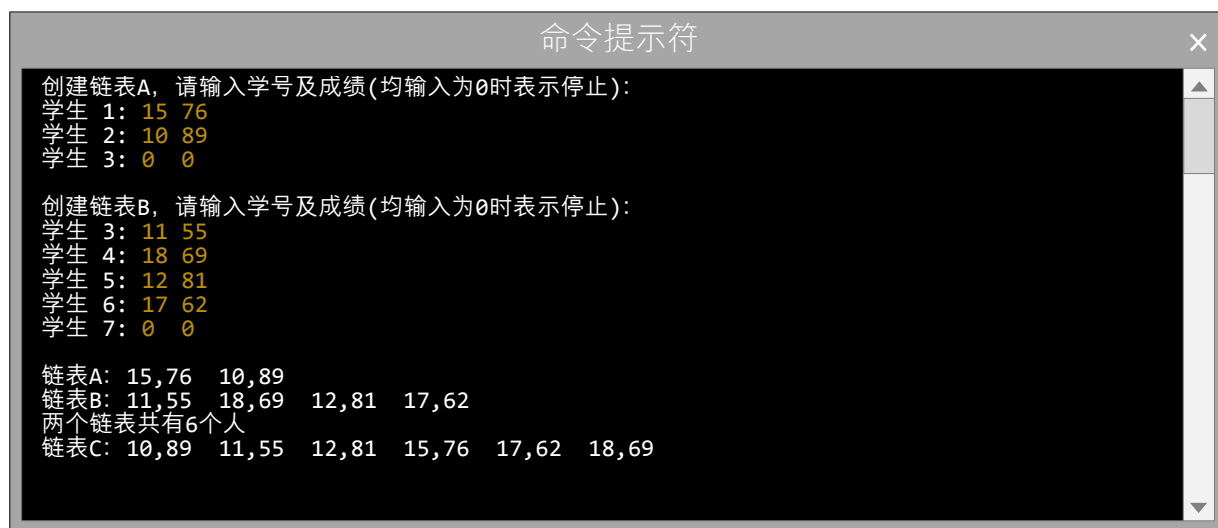



图 786.1

附件： P786.c

文件内容：

P786.c

```

#include<stdio.h>
#include<malloc.h>

#define LEN sizeof(struct student)

int sum = 0;
/* User Code Begin(考生可在本行后添加代码，定义程序中使用的结构体类型、声明自定义函数的原型，行数不限) */
<A>

```

```

/* User Code End(考生添加代码结束) */

/* print以规定的格式完成遍历显示指定的链表 */
void print(struct student *Head);

int main(void)
{
    struct student *ah, *bh, *ac;

    printf("创建链表A, 请输入学号及成绩(均输入为0时表示停止): \n");
    ah = creat();
    printf("\n创建链表B, 请输入学号及成绩(均输入为0时表示停止): \n");
    bh = creat();

    printf("\n链表A: ");
    print(ah);
    printf("\n链表B: ");
    print(bh);

    ac = merge(ah, bh);
    printf("\n两个链表共有%d个人\n链表C: ", sum);
    print(ac);

    return 0;
}

void print(struct student *Head)
{
    while (Head != NULL)
    {
        printf("%d,%d ", Head->num, Head->score);
        Head = Head->next;
    }
}

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */
<B>

```

参考答案:

```

<A> struct student
{
    int num, score;
    struct student *next;
};

struct student *creat(void);
struct student *merge(struct student *ah, struct student *bh);
<B> struct student *creat(void)
{
    struct student *Head, *p1, *p2;
    int flag = 0;

    Head = p1 = p2 = (struct student*)malloc(LEN);
    while (1)
    {
        sum++;
        printf("学生 %d: ", sum);
        scanf("%d%d", &p1->num, &p1->score);
        if (p1->num == 0 && p1->score == 0)

```

```

        {
            break;
        }

        if (Head == NULL)
        {
            Head = p1;
        }
        else
        {
            p2->next = p1;
        }
        p2 = p1;
        p1 = (struct student *)malloc(LEN);
        flag = 1;
    }
    sum--;
    p2->next = NULL;

    if (flag == 0)
    {
        return NULL;
    }
    return Head;
}

struct student *merge(struct student *ah, struct student *bh)
{
    int i, tmpnum, tmpscore;
    struct student *Head, *p0;

    if (ah == NULL && bh == NULL)
    {
        return NULL;
    }
    else if (ah != NULL && bh == NULL)
    {
        Head = ah;
    }
    else if (ah == NULL && bh != NULL)
    {
        Head = bh;
    }
    else
    {
        Head = p0 = ah;
        while (p0->next != NULL)
        {
            p0 = p0->next;
        }
        p0->next = bh;
    }

    for (i = 1; i < sum; i++)
    {
        p0 = Head;
        while (p0->next != NULL)
        {
            if (p0->num > p0->next->num)
            {
                tmpnum = p0->num;
                p0->num = p0->next->num;
                p0->next->num = tmpnum;
            }
        }
    }
}

```



```

        tmpscore = p0->score;
        p0->score = p0->next->score;
        p0->next->score = tmpscore;
    }
    p0 = p0->next;
}
}

return Head;
}

```

P787

DIFFICUTLY ★★

程序 P787.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。


程序的功能是：从键盘输入 2 个字符串（约定每个字符串中字符数 ≤ 80 字节），将此 2 个字符串连接成一个新字符串并输出。

注：程序中不能使用库函数 `strcpy()`、`strcat()`、`strncat()`、`strncpy()`、`memcpy()`、`strncpy()` 或使用同名的变量、函数、单词。

程序的运行效果应类似地如图 787.1 所示，金色部分是从键盘输入的内容。



图 787.1

附件:  P787.c

文件内容:

P787.c

```

#include<stdio.h>

#define MAXLINE 80

/* userCode(<80字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    char str[2][MAXLINE+1], strall[2*MAXLINE+1]="", *pNew;

    printf("input 2 strings:\n");
    gets(str[0]);
    gets(str[1]);

    pNew = strLianjie(strLianjie(strall, str[0]), str[1]);
    printf("\nResult: %s\n", pNew);

    return 0;
}

```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<B>
```

参考答案:


```
<A> char *strLianjie(char str1[], char str2[]);
<B> char *strLianjie(char str1[], char str2[])
{
    int i;
    static int j;

    for (i = 0; str2[i] != '\0'; i++, j++)
    {
        str1[j] = str2[i];
    }
    str1[j] = '\0';

    return str1;
}
```

P788

DIFFICUTLY ★★

程序 P788.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：先从 main() 函数中输入数组长度 n （约定 $n \leq 20$ ），再调用自定义函数 scanfArr() 完成数组中的每个元素读入，然后分别调用自定义函数 maxArr()、aver() 计算数组元素的最大值、平均值，最后输出最大值、平均值。

注：要求用指针完成函数中数组参数的传递、以及各个数组元素的访问，即自定义函数头和函数体中不得出现数组下标形式的表示法。

程序的运行效果应类似地如图 788.1 所示，金色部分是从键盘输入的内容。

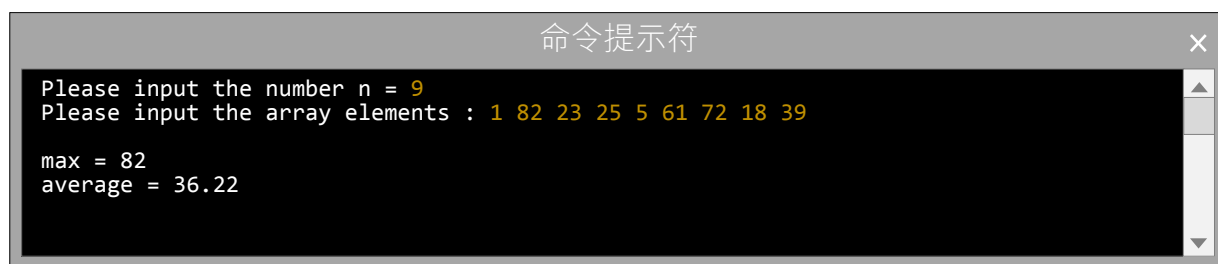



图 788.1

附件:  P788.c

文件内容:

P788.c

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

```
/* 本部分代码功能建议: 函数原型声明 */
```

```
/* User Code Begin(Limit: lines<=3, lineLen<=80, 考生可在本行后添加代码、最多3行、行长<=80字符) */
```

```
<A>
```

/* User Code End(考生添加代码结束。注意：空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

```
int main(void)
{
    int Data[20], n, max;
    double average;

    printf("Please input the number n = ");
    scanf("%d", &n);
    printf("Please input the array elements : ");
    scanfArr(Data, n);

    max = maxArr(Data, n);
    average = aver(Data, n);
    printf("\nmax = %d\naverage = %.2f\n", max, average);

    return 0;
}
```

/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */

参考答案:

```
<A> void scanfArr(int *Data, int n);
    int maxArr(int *Data, int n);
    double aver(int *Data, int n);
<B> void scanfArr(int *Data, int n)
{
    int i;

    for (i = 0; i < n; i++)
    {
        scanf("%d", Data + i);
    }
}

int maxArr(int *Data, int n)
{
    int max = *(Data + 0), i;

    for (i = 1; i < n; i++)
    {
        if (*(Data + i) > max)
        {
            max = *(Data + i);
        }
    }

    return max;
}

double aver(int *Data, int n)
{
    int sum = 0, i;
    double average;

    for (i = 0; i < n; i++)
    {
        sum += *(Data + i);
    }
}
```

```

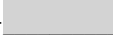
        average = sum / (float)n;

    return average;
}

```

P790

DIFFICUTLY ★★

程序 P790.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：从键盘上读入 5 个学生的姓名（char(10)）、学号（char(10)）和成绩（int），要求成绩必须在 0 ~ 100 之间，否则重新读入该学生信息。最后输出不及格学生的学号、名字和分数。

注：要求用指针完成函数中结构体数组参数的传递以及各个数组元素的访问，访问结构体成员时使用“->”形式，自定义函数头和函数体中不得出现数组下标形式的表示法。

可用素材：

```

printf("input name number score:\n")
printf("student %d: "...
printf("          error score! input again!\n")
printf("%s/%s,%d  "...

```

程序的运行效果应类似地如图 790.1 所示，金色部分是从键盘输入的内容。



图 790.1

附件:  P790.c

文件内容:

P790.c

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

```
/* User Code Begin(考生可在本行后添加代码，例如结构体的定义、函数原型声明等，行数不限) */
```

```
<A>
```

```
/* User Code End(考生添加代码结束) */
```

```

int main(void)
{
    struct stu stud[5];

    input(stud, 5);
    printf("\nfailed the exam: ");
    output(stud, 5);

    return 0;
}

```

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */

参考答案:

```
<A> struct stu
{
    char name[10], num[10];
    int score;
};

void input(struct stu *stud, int n);
void output(struct stu *stud, int n);
<B> void input(struct stu *stud, int n)
{
    int i, count = 0;

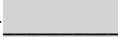
    printf("input name number score:\n");
    for (i = 0; i < n; i++)
    {
        do
        {
            count++;
            printf("student %d: ", count);
            scanf("%s%s%d", (stud + i)->name, (stud + i)->num, &(stud + i)->score);
            if ((stud + i)->score < 0 || (stud + i)->score > 100)
            {
                printf("          error score! input again!\n");
                count--;
            }
        } while ((stud + i)->score < 0 || (stud + i)->score > 100);
    }
}

void output(struct stu *stud, int n)
{
    int i, flag = 0;

    for (i = 0; i < n; i++)
    {
        if ((stud + i)->score < 60)
        {
            printf("%s/%s,%d  ", (stud + i)->num, (stud + i)->name, (stud + i)->score);
        }
        flag = 1;
    }
    if (flag == 0)
    {
        printf("Not Find!");
    }
    putchar('\n');
}
```

P791

DIFFICUTLY ★★

程序 P791.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：先从键盘上读入节点数 n ，再调用自定义函数 creatlink() 从键盘读入每个节点的数据、建立链表，接着调用自定义函数 printlink() 输出链表内容，然后调用自定义函数 delodd() 实现删除所有的值为奇数的节点，最后调用自定义函数 printlink() 输出链表中余下的节点。

可用素材： printf("Please input the data : ")

程序的运行效果应类似地如图 791.1 所示，金色部分是从键盘输入的内容。

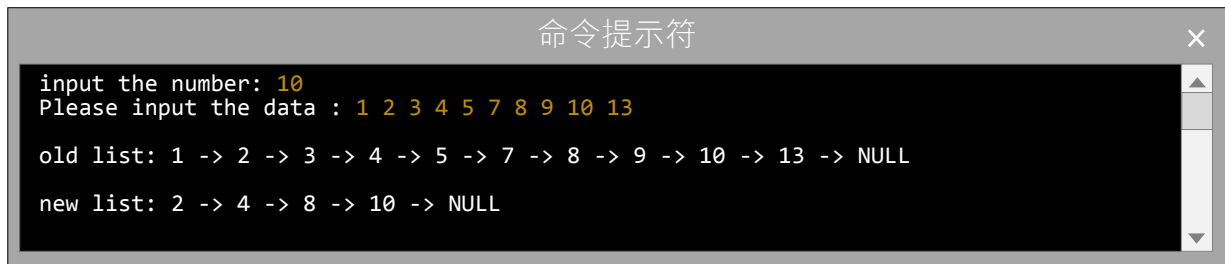



图 791.1

附件:  P791.c

文件内容:

P791.c

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

/* User Code Begin(考生可在本行后添加代码，定义程序中使用的结构体类型、声明自定义函数的原型，行数不限) */
<A>
/* User Code End(考生添加代码结束) */

/* print以规定的格式完成遍历显示指定的链表 */
void printlink(struct link *Head);

int main(void)
{
    struct link *Head;
    int n;

    printf("input the number: ");
    scanf("%d", &n);

    Head = creatlink(n);
    printf("\nold list: ");
    printlink(Head);

    Head = delodd(Head);
    printf("\nnew list: ");
    printlink(Head);

    return 0;
}

void printlink(struct link *Head)
```

```

{
    while (Head != NULL)
    {
        printf("%d -> ", Head->data);
        Head = Head->next;
    }
    puts("NULL");
}

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */
<B>

```

参考答案:

```

<A> struct link
{
    int data;
    struct link *next;
};

struct link *creatlink(int n);
struct link *delodd(struct link *Head);
<B> struct link *creatlink(int n)
{
    struct link *Head, *p1, *p2;
    int i;

    if (n == 0)
    {
        return NULL;
    }

    Head = p1 = p2 = (struct link *)malloc(sizeof(struct link));
    printf("Please input the data : ");
    for (i = 1; i <= n; i++)
    {
        scanf("%d", &p1->data);
        if (Head == NULL)
        {
            Head = p1;
        }
        else
        {
            p2->next = p1;
        }
        p2 = p1;
        p1 = (struct link *)malloc(sizeof(struct link));
    }
    p2->next = NULL;

    return Head;
}

struct link *delodd(struct link *Head)
{
    struct link *p1, *p2;

    if (Head == NULL)
    {
        return Head;
    }
}

```

```

    p1 = Head->next;
    p2 = Head;
    while (p1 != NULL)
    {
        if (p1->data % 2 != 0)
        {
            p2->next = p1->next;
            free(p1);
            p1 = p2->next;
        }
        else
        {
            p2 = p1;
            p1 = p1->next;
        }
    }
    if (Head->data % 2 != 0)
    {
        p2 = Head;
        Head = Head->next;
        free(p2);
    }

    return Head;
}

```

P795

DIFFICUTLY ★★

程序 P795.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：一个长度不超过 199 的字符串，字符串中只含字母和空格，空格用于分隔单词。请将字符串中用空格分隔的单词在屏幕上输出来。

注：要求用指针完成函数中各参数的传递与访问，自定义函数头和函数体中不得出现数组下标形式的表示法。

可用素材： `printf("Please input the data : ")`

提示： 利用指针数组记录每个单词的开始位置，把字符串中单词结束后的空格改为 `'\0'`。

程序的运行效果应类似地如图 795.1 所示，金色部分是从键盘输入的内容。



图 795.1

附件：
P795.c

文件内容：

P795.c

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

```
/* userCode(<80字符): 自定义函数之原型声明 */
```

```
<A>
```

```
int main(void)
{
    char str[200], *pStr[101];
    int i=0, count;

    printf("Please input a string: ");
    gets(str);

    count = split(str, pStr);
    printf("\n%d Words: ", count);
    for (i=0; i<count; i++)
    {
        printf("%s-", pStr[i]);
    }
    putchar('\n');

    return 0;
}
```

```
/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
```

```
<B>
```

参考答案:

```
<A> int split(char *str, char **pStr);
<B> int split(char *str, char **pStr)
{
    int count = 0, m0 = 0;

    while (*str != '\0')
    {
        if (*str == ' ')
        {
            if (m0 != 0)
            {
                *str = 0;
                m0 = 0;
            }
        }
        else if (m0 == 0)
        {
            m0 = 1;
            *(pStr + count) = str;
            count++;
        }
        str++;
    }

    return count;
}
```

P797
DIFFICUTLY ★★

程序 P797.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码

或将  换成代码）。

程序的功能是：从键盘读入 5 行 9 列整数保存到二维数组中，调用用户自定义函数查找数组中最大元素（约定只考虑仅有一个最大的情况）及其所在位置的行下标、列下标。

程序的运行效果应类似地如图 797.1 所示，金色部分是从键盘输入的内容。

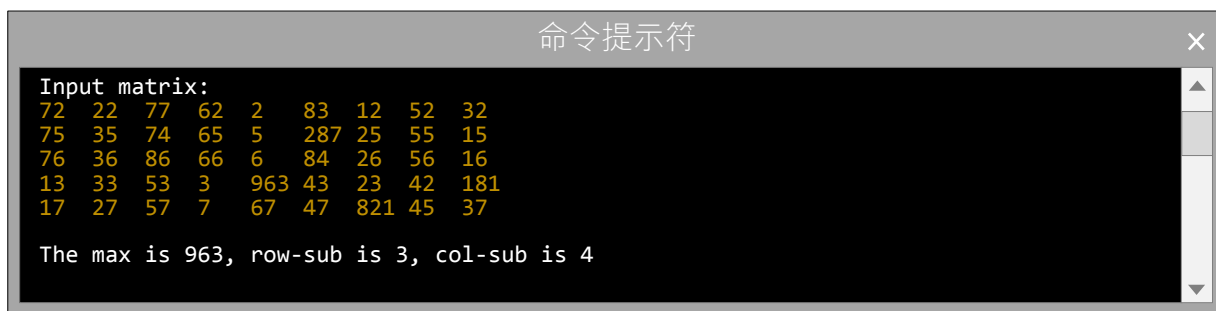


图 797.1

附件:  P797.c

文件内容:

P797.c

```
#include <stdio.h>

/* userCode(<80字符>): 自定义函数之原型声明 */
<A>

int main(void)
{
    int array[5][9], i, j, max, maxRow, maxCol;

    printf("Input matrix:\n");
    for (i=0; i<5; i++)
    {
        for (j=0; j<9; j++)
        {
            scanf("%d", &array[i][j]);
        }
    }

    /* userCode(<80字符>): 调用函数查找数组中最大元素及其所在位置的行下标、列下标 */
    <B>
    printf("\nThe max is %d, row-sub is %d, col-sub is %d\n", max, maxRow, maxCol);

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计，行数不限 */
<C>
```

参考答案:

```
<A> void search(int (*array)[9], int *pmax, int *pmaxRow, int *pmaxCol);
<B> search(array, &max, &maxRow, &maxCol);
<C> void search(int (*array)[9], int *pmax, int *pmaxRow, int *pmaxCol)
{
    int i, j;

    *pmax = array[0][0];
    *pmaxRow = 0;
```

```

    *pmaxCol = 0;
    for (i = 0; i < 5; i++)
    {
        for (j = 0; j < 9; j++)
        {
            if (array[i][j] > *pmax)
            {
                *pmax = array[i][j];
                *pmaxRow = i;
                *pmaxCol = j;
            }
        }
    }
}

```

P798

DIFFICUTLY ★★


程序 P798.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：已知一学生信息结构体，从键盘输入一个整数 m ，调用子函数 `alloc_memery()` 动态分配 m 个结构体存储空间，并将每个结构体的学号成员变量依次初始化为 $1 \sim m$ 。

程序的运行效果应类似地如图 798.1 所示，金色部分是从键盘输入的内容。



图 798.1

附件:  P798.c

文件内容:

P798.c

```

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

typedef struct STUDENT
{
    int studentID;           //学号
    char studentName[10];    //姓名
    int scoreComputer;       //计算机成绩
} STUD;

/* userCode(<80字符): 自定义函数之原型声明 */
<A>

int main(void)
{
    int m, i;
    STUD *pStu;

    printf("Input m: ");
    scanf("%d", &m);

```

```

pStu = alloc_memery(m);
if (pStu != NULL)
{
    printf("\n分配成功, 学号依次被初始化为: ");
    for (i=0; i<m; i++)
    {
        printf("%3d", pStu[i].studentID);
    }
    free(pStu);
}
else
{
    printf("\n分配失败!\n");
}

return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */
<B>

```

参考答案:

```

<A> STUD *alloc_memery(int m);
<B> STUD *alloc_memery(int m)
{
    STUD *pStu;
    int i;

    pStu = (STUD *)malloc(m * sizeof(STUD));
    if (pStu == NULL)
    {
        return pStu;
    }


    for (i = 0; i < m; i++)
    {
        (pStu + i)->studentID = i + 1;
    }

    return pStu;
}

```

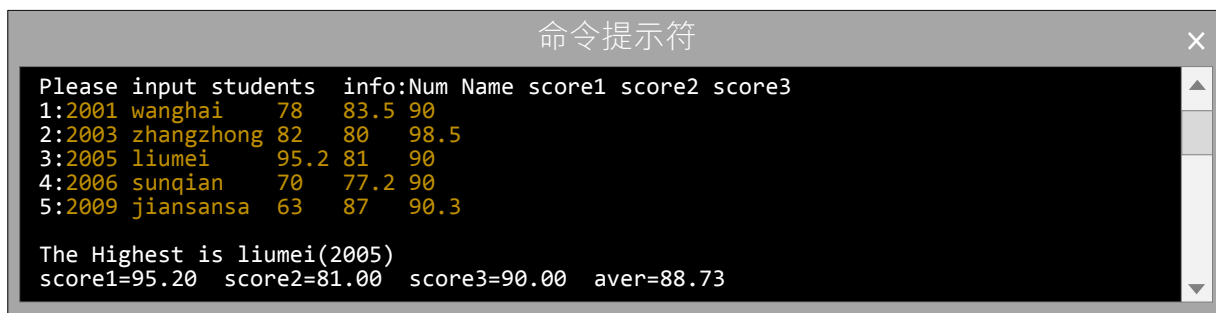
P801

DIFFICUTLY ★★

程序 P801.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：有五个学生，每个学生的数据包括学号、姓名（最长 19 字节）、三门课的成绩，从键盘输入五个学生的数据，并计算每个学生的平均成绩，最后显示最高平均分的学生的信息（包括学号，姓名，三门课的成绩，平均分）。

程序的运行效果应类似地如图 801.1 所示，金色部分是从键盘输入的内容。



```
命令提示符
Please input students info:Num Name score1 score2 score3
1:2001 wanghai 78 83.5 90
2:2003 zhangzhong 82 80 98.5
3:2005 liumei 95.2 81 90
4:2006 sunqian 70 77.2 90
5:2009 jiansansa 63 87 90.3

The Highest is liumei(2005)
score1=95.20 score2=81.00 score3=90.00 aver=88.73
```

图 801.1

附件:  P801.c

文件内容:

P801.c

```
#include <stdio.h>

struct student
{
    int num;
    char name[20];
    float score1, score2, score3;
    float aver;
};

void Input(struct student *pStu, int n);
int Highest(struct student *pStu, int n);

int main(void)
{
    int high; /* high记录平均分最高的学生的序号，具体使用参考后面的代码 */
    struct student myClass[5];

    /* 本部分代码功能建议：调用相应的函数完成数据输入和统计 */
    /* User Code Begin(Limit: lines<=2, lineLen<=50, 考生可在本行后添加代码、最多2行、行长<=50字符) */
    <A>
    /* User Code End(考生添加代码结束。注意：空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

    printf("\nThe Highest is %s(%d)\nscore1=%.2f score2=%.2f score3=%.2f aver=%.2f\n",
        myClass[high].name, myClass[high].num,
        myClass[high].score1, myClass[high].score2, myClass[high].score3, myClass[high].aver);

    return 0;
}

/* 输入N个学生的信息并计算平均分 */
void Input(struct student *pStu, int n)
{
    int i;
    struct student tmpStu;

    printf("Please input students info:Num Name score1 score2 score3\n");
    for (i=0; i<n; i++)
    {
        printf("%d:", i+1);
        scanf("%d%s%f%f%f", &tmpStu.num, tmpStu.name,
            &tmpStu.score1, &tmpStu.score2, &tmpStu.score3);
        pStu[i] = tmpStu;
    }
}
```

```

        pStu[i].aver = (pStu[i].score1 + pStu[i].score2 + pStu[i].score3) / 3.0f;
    }
}

/* 找出并通过函数值返回最高平均分学生的序号 */
int Highest(struct student *pStu, int n)
{
    int i, pos = 0;

    /* User Code Begin(考生可在本行后添加代码, 行数不限) */
    <B>
    /* User Code End(考生添加代码结束) */

    return pos;
}

```

参考答案:

```


<A> Input(myClass, 5);
    high = Highest(myClass, 5);
<B> float max = pStu[0].aver;

    for (i = 1; i < n; i++)
    {
        if (pStu[i].aver > max)
        {
            max = pStu[i].aver;
            pos = i;
        }
    }
}

```

P807

DIFFICUTLY ★★

程序 P807.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的功能是：有五个学生，每个学生的数据包括学号、姓名（最长 19 字节）、三门课的成绩，从键盘输入五个学生的数据，并计算每个学生的平均成绩，最后显示最高平均分的学生的信息（包括学号，姓名，三门课的成绩，平均分数）。

程序的运行效果应类似地如图 807.1 所示，金色部分是从键盘输入的内容。

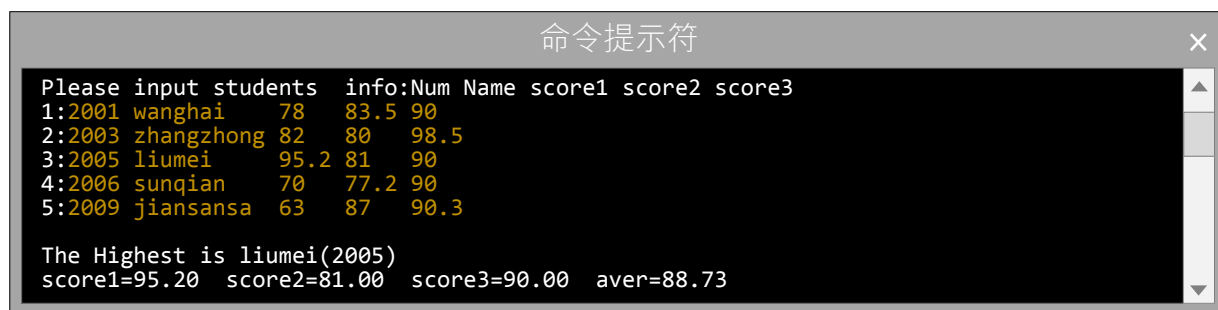


图 807.1

附件:  P807.c

文件内容:

P807.c

```
#include <stdio.h>

/* User Code Begin(考生可在本行后添加代码, 定义程序中使用的数据类型SCORE, 行数不限) */
<A>
/* User Code End(考生添加代码结束) */

void Input(SCORE *pStu, int n);
int Highest(SCORE *pStu, int n);

int main(void)
{
    int high; /* high记录平均分最高的学生的序号, 具体使用参考后面的代码 */
    SCORE myClass[5];

    /* 本部分代码功能建议: 调用相应的函数完成数据输入和统计 */
    /* User Code Begin(Limit: lines<=2, lineLen<=50, 考生可在本行后添加代码、最多2行、行长<=50字符) */
    <B>
    /* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

    printf("\nThe Highest is %s(%d)\nscore1=%.2f score2=%.2f score3=%.2f aver=%.2f\n",
           myClass[high].name, myClass[high].num,
           myClass[high].score1, myClass[high].score2, myClass[high].score3, myClass[high].aver);

    return 0;
}

/* 输入N个学生的信息并计算平均分 */
void Input(SCORE *pStu, int n)
{
    int i;
    SCORE tmpStu;

    printf("Please input students info:Num Name score1 score2 score3\n");
    for (i=0; i<n; i++)
    {
        printf("%d:", i+1);
        scanf("%d%s%f%f%f", &tmpStu.num, tmpStu.name,
              &tmpStu.score1, &tmpStu.score2, &tmpStu.score3);
        pStu[i] = tmpStu;
        pStu[i].aver = (pStu[i].score1 + pStu[i].score2 + pStu[i].score3) / 3.0f;
    }
}

/* 找出并通过函数值返回最高平均分学生的序号 */
int Highest(SCORE *pStu, int n)
{
    int i, pos = 0;

    /* User Code Begin(考生可在本行后添加代码, 行数不限) */
    <C>
    /* User Code End(考生添加代码结束) */

    return pos;
}
```

参考答案:

```
<A> typedef struct student
{
    char name[20];
    int num;
    float score1, score2, score3, aver;
} SCORE;
<B> Input(myClass, 5);
    high = Highest(myClass, 5);
<C> float max = pStu[0].aver;

    for (i = 1; i < n; i++)
    {
        if (pStu[i].aver > max)
        {
            max = pStu[i].aver;
            pos = i;
        }
    }
```

P826

DIFFICUTLY ★★

程序 P826.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：

1. 从键盘读入一行字符 **Str**（其中可以有空格，约定字符数 ≤ 127 字节）和整数 m 。若 **Str** 长度为 0，则报错并重新读取，直到符合要求为止；若 m 小于 1 或 m 大于 **Str** 的长度，则报错并重新读取，直至符合要求为止。这一功能由函数 getInput() 实现。
2. 将字符串 **Str** 中从第 m 个字符开始的全部字符复制到 **dstStr** 中。这一功能由函数 copyStr() 实现。

注：程序中不能使用库函数 strcpy()、strcat()、strncat()、strncpy()、memcpy()、strncpy() 或使用同名的变量、函数、单词。

可用素材：

```
printf("please input a string: ")
printf("\nError Str, please input a string again: ")
printf("please input m: ")
printf("\nError m, please input m again: ")
printf("Input is: Str=... m=..."...)
printf("Result is: ..."...)
```

程序的运行效果应类似地如图 826.1 和图 826.2 所示，金色部分是从键盘输入的内容。




图 826.1 输入“abcdefgh”和“3”时



```
命令提示符
please input a string:
Error Str, please input a string again: abc
please input m: 0
Error m, please input m again: 8
Error m, please input m again: 3
Input is: Str=abc  m=3
Result is: c
```

图 826.2 输入空行（直接回车）、“abc” “0” “8” 和 “3” 时

附件:  P826.c

文件内容:

P826.c

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

/* 本部分代码功能建议：函数原型声明 */
/* User Code Begin(Limit: lines<=2, lineLen<=80, 考生可在本行后添加代码、最多2行、行长<=80字符) */
<A>

/* User Code End(考生添加代码结束。注意：空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

int main(void)
{
    int m, i;
    char Str[128], dstStr[128];

    m = getInput(Str);
    printf("\nInput is: Str=%s  m=%d\n", Str, m);

    for (i=0; i<128; i++)
    {
        dstStr[i] = '\0';
    }
    copyStr(Str, m, dstStr);
    printf("\nResult is: %s\n", dstStr);

    return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分，如函数的定义，行数不限) */
<B>
```

参考答案:

```
<A> int getInput(char Str[]);
    void copyStr(char Str[], int m, char dstStr[]);
<B> int getInput(char Str[])
{
    int len, m;

    printf("please input a string: ");
    gets(Str);
```

```

len = strlen(Str);
while (len == 0)
{
    printf("\nError Str, please input a string again: ");
    gets(Str);
    len = strlen(Str);
}

printf("please input m: ");
scanf("%d", &m);
while (m < 1 || m > len)
{
    printf("\nError m, please input m again: ");
    scanf("%d", &m);
}

return m;
}

void copyStr(char Str[], int m, char dstStr[])
{
    int i, j;

    for (i = m - 1, j = 0; Str[i] != '\0'; i++, j++)
    {
        dstStr[j] = Str[i];
    }
    dstStr[j] = '\0';
}

```

P317

DIFFICUTLY ★★

根据输入的源文件名（含路径，< 100 字节）和目标文件名（含路径，< 100 字节），实现将源文件复制到目标文件。注意事项：

1. 源文件可能是文本文件，也可能是二进制文件。
2. 程序的返回值（即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值，也称退出代码）规定为：
 - a) 复制成功，返回 0。
 - b) 源文件打开失败，返回 2。
 - c) 目标文件创建失败，返回 3。
 - d) 向目标文件写数据的过程中出错，返回 4。
3. 向目标文件写数据的过程中出错的情况很少发生，用户根据图例中的输入数据进行测试时，很可能不会出错，但程序应考虑出错的情况（例如磁盘空间不够、往 U 盘上写一个大文件的过程中 U 盘出错或被拔走）。

可用素材：

```

printf("Please input sourceFilename: ")
printf("Please input destinationFilename: ")
printf("\ncopy %s to %s succeeded!\n"...
printf("\nsource File (%s) Open Error!\n"...
printf("\ndestination File (%s) Create Error!\n"...
printf("\nwriting destination File (%s) Error!\n"...

```

程序的运行效果应类似地如图 317.1、图 317.2、图 317.3 和图 317.4 所示，金色部分是从键盘输入的内容。



图 317.1 源文件存在，目标文件创建成功，复制正常完成



图 317.2 源文件打开失败



图 317.3 目标文件创建失败



图 317.4 向目标文件写数据的过程中出错

参考答案:

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

```
int main(void)
```

```
{
```

```
    FILE *fp1, *fp2;
```

```
    char source[100], destination[100];
```

```
    int ch;
```

```
    printf("Please input sourceFilename: ");
```

```
    gets(source);
```

```
    printf("Please input destinationFilename: ");
```

```
    gets(destination);
```

```
    fp1 = fopen(source, "rb");
```

```
    if (fp1 == NULL)
```

```
    {
```

```
        printf("\nsource File (%s) Open Error!\n", source);
```

```
        return 2;
```

```
    }
```

```

fp2 = fopen(destination, "wb");
if (fp2 == NULL)
{
    printf("\ndestination File (%s) Create Error!\n", destination);
    fclose(fp1);
    return 3;
}

while (!feof(fp1))
{
    ch = fgetc(fp1);
    if (ch != EOF)
    {
        if (fputc(ch, fp2) == EOF)
        {
            printf("\nwriting destination File (%s) Error!\n", destination);
            fclose(fp1);
            fclose(fp2);
            return 4;
        }
    }
}
printf("\ncopy %s to %s succeeded!\n", source, destination);
fclose(fp1);
fclose(fp2);
return 0;
}

```

P312

DIFFICUTLY ★★

有一存储很多商品数据（每件商品的属性先后包括：品名、规格、数量、单价，编程时相应的数据类型分别定义为字符串 char (20)、字符串 char (12)、long、float）的二进制文件“sp.dat”（即未作任何格式转换而直接使用 fwrite() 将商品属性写入文件），从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp.dat”（见附件）保存到程序 P312.c 所在的文件夹且文件名保持不变。

可用素材： printf("Please input shang pin pin ming:")
printf("\ncha zhao qing kuang:\n")
printf("mei you shang pin :...")

程序的运行效果应类似地如图 312.1 和图 312.2 所示，**金色部分**是从键盘输入的内容。

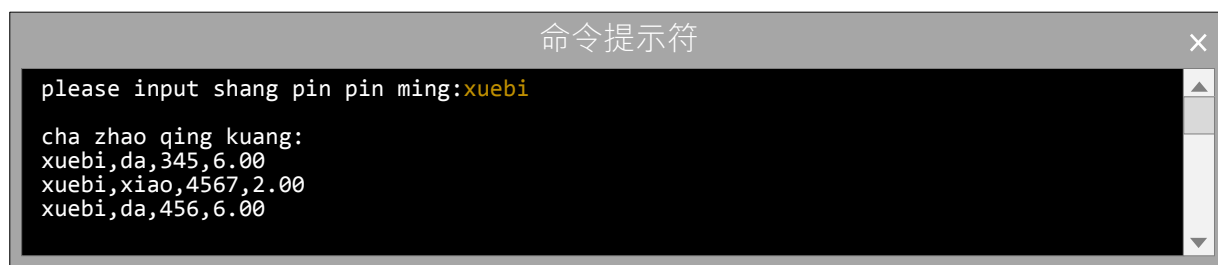


图 312.1 输入“xuebi”时

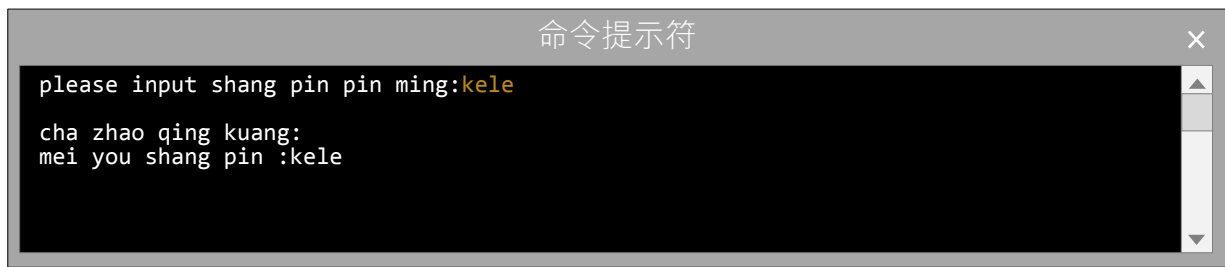


图 312.2 输入“kele”时

附件:  sp.dat

测试内容备选: xuebi、xianchengduo

参考答案:

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

```
int main(void)
{
    FILE *fp;
    char name[20], size[12], search[20];
    long qty;
    float price;
    int flag = 0;

    printf("Please input shang pin pin ming:");
    gets(search);

    fp = fopen("sp.dat", "rb");
    if (fp == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    printf("\ncha zhao qing kuang:\n");
    while (!feof(fp))
    {
        if (fread(name, 1, 20, fp) == 20 && fread(size, 1, 12, fp) == 12 &&
            fread(&qty, 4, 1, fp) == 1 && fread(&price, 4, 1, fp) == 1)
        {
            if (strcmp(name, search) == 0)
            {
                printf("%s,%s,%d,%.2f\n", name, size, qty, price);
                flag = 1;
            }
        }
    }
    if (flag == 0)
    {
        printf("mei you shang pin :%s\n", search);
    }

    fclose(fp);

    return 0;
}
```

P314

DIFFICUTLY ★★

有一存储很多商品数据（每件商品的属性先后包括：品名、规格、单价（有小数位）、数量，数据的最长长度分别为 20、10、6、5，在文件中以空格为分隔，每个商品的数据占一行）的文本文件，从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔，单价显示时只显示 2 位小数），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp.txt”（见附件及文件内容）保存到程序 P314.c 所在的文件夹且文件名保持不变。

可用素材：

```
printf("Please input shang pin pin ming:")
printf("\ncha zhao qing kuang:\n")
printf("mei you shang pin :...
```

程序的运行效果应类似地如图 314.1 和图 314.2 所示，金色部分是从键盘输入的内容。

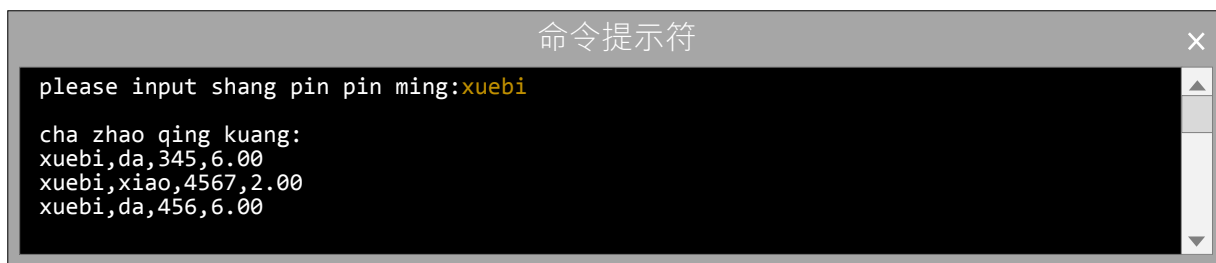


图 314.1 输入“xuebi”时



图 314.2 输入“kele”时

附件： 
sp.txt

文件内容：

sp.txt

```
xuebi da 6.00 345
nongfuSQxianchengduo zhongxingA 4.392 57398
xuebi xiao 2.004 4567
xuebi da 6.003 456
```

参考答案：

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

int main(void)
{
    FILE *fp;
    char name[21], size[11], search[21];
    float price;
    int qty, flag = 0;
```

```

printf("Please input shang pin pin ming:");
gets(search);

fp = fopen("sp.txt", "r");
if (fp == NULL)
{
    printf("File open error!\n");
    return 0;
}

printf("\ncha zhao qing kuang:\n");
while (!feof(fp))
{
    if (fscanf(fp, "%s%s%f%d", name, size, &price, &qty) == 4)
    {
        if (strcmp(name, search) == 0)
        {
            printf("%s,%s,%d,%.2f\n", name, size, qty, price);
            flag = 1;
        }
    }
}
if (flag == 0)
{
    printf("mei you shang pin :%s\n", search);
}

fclose(fp);

return 0;
}

```

P318

DIFFICUTLY ★★

有一存储很多商品数据（每件商品的属性先后包括：品名、规格、数量、单价，编程时相应的数据类型分别定义为字符串 char (18)、字符串 char (12)、long、float）的二进制文件“sp38.dat”（即未作任何格式转换而直接使用 fwrite() 将商品属性写入文件），从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp38.dat”（见附件）保存到程序 P318.c 所在的文件夹且文件名保持不变。

可用素材： printf("Please input shang pin pin ming:")
 printf("\ncha zhao qing kuang:\n")
 printf("mei you shang pin :...")


提示： 对于结构体方法，使用 fread() 与 sizeof(结构体) 肯定不对，应单项属性逐个 fread()。程序的运行效果应类似地如图 318.1 和图 318.2 所示，金色部分是从键盘输入的内容。

```
命令提示符
please input shang pin pin ming:xuebi
cha zhao qing kuang:
xuebi,da,345,6.00
xuebi,xiao,4567,2.00
xuebi,da,456,6.00
```

图 318.1 输入“xuebi”时

```
命令提示符
please input shang pin pin ming:kele
cha zhao qing kuang:
mei you shang pin :kele
```

图 318.2 输入“kele”时

附件:  sp38.dat

测试内容备选: xuebi、xianchengduo

参考答案:

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

int main(void)
{
    FILE *fp;
    char name[18], size[12], search[18];
    long qty;
    float price;
    int flag = 0;

    printf("Please input shang pin pin ming:");
    gets(search);

    fp = fopen("sp38.dat", "rb");
    if (fp == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    printf("\ncha zhao qing kuang:\n");
    while (!feof(fp))
    {
        if (fread(name, 1, 18, fp) == 18 && fread(size, 1, 12, fp) == 12 &&
            fread(&qty, 4, 1, fp) == 1 && fread(&price, 4, 1, fp) == 1)
        {
            if (strcmp(name, search) == 0)
            {
                printf("%s,%s,%d,%.2f\n", name, size, qty, price);
                flag = 1;
            }
        }
    }
    if (flag == 0)
    {
        printf("mei you shang pin :%s\n", search);
    }
}
```



```

    }

    fclose(fp);

    return 0;
}

```

P319

DIFFICULTLY ★★

有一存储很多商品数据的二进制文件“sp36.dat”，每件商品的属性先后包括：品名（17 字节的字符串）、规格（12 字节的字符串）、数量（3 字节的整数）、单价（float 实数）。从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp36.dat”（见附件）保存到程序 P319.c 所在的文件夹且文件名保持不变。

可用素材： `printf("Please input shang pin pin ming:");`
`printf("\ncha zhao qing kuang:\n");`
`printf("mei you shang pin :...");`

提示： 对于结构体方法，使用 `fread()` 与 `sizeof(结构体)` 肯定不对，应单项属性逐个 `fread()`。程序的运行效果应类似地如图 319.1 和图 319.2 所示，金色部分是从键盘输入的内容。

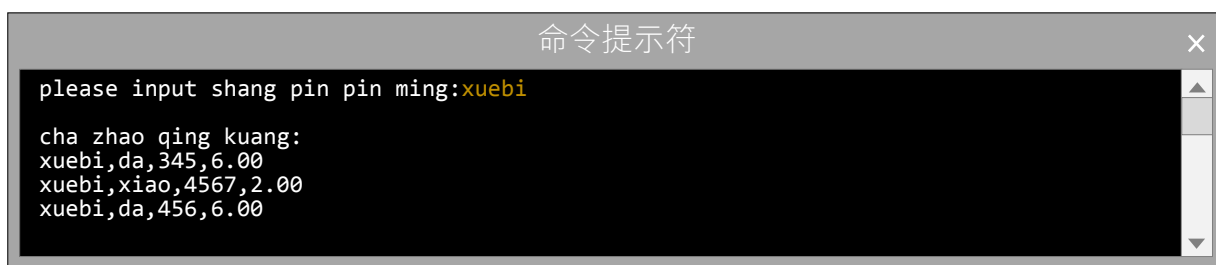



图 319.1 输入“xuebi”时



图 319.2 输入“kele”时

附件：  sp36.dat

测试内容备选： xuebi、xianchengduo

参考答案：

```

#include <stdio.h>
#include <string.h>

int main(void)
{
    FILE *fp;
    char name[17], size[12], search[17];
    float price;

```

```

int qty = 0, flag = 0;    /* 另解: 可对 qty 不进行初始化赋值 */

printf("Please input shang pin ming:");
gets(search);

fp = fopen("sp36.dat", "rb");
if (fp == NULL)
{
    printf("File open error!\n");
    return 0;
}

printf("\ncha zhao qing kuang:\n");
while (!feof(fp))
{
    if (fread(name, 1, 17, fp) == 17 && fread(size, 1, 12, fp) == 12 &&
        fread(&qty, 3, 1, fp) == 1 && fread(&price, 4, 1, fp) == 1)
    {
        if (strcmp(name, search) == 0)
        {
            printf("%s,%s,%d,%.2f\n", name, size, qty, price);
            flag = 1;
        }    /* 对应另解, qty 的输出表列为 qty & 0x00ffffff */
    }
}
if (flag == 0)
{
    printf("mei you shang pin :%s\n", search);
}

fclose(fp);

return 0;
}

```

P322

DIFFICUTLY ★★

在文本文件“Comp.txt”里有需要计算结果的整数算式，每个算式占一行且文件中有多个算式，运算类型只有“加法(+)”或者“减法(-)”且运算符前后至少有一个空格——但其中可能有空行和不符合要求的算式（但其行长肯定不超过 200 字节）。计算这些算式的结果并在屏幕上显示，空行不作任何处理，不符合要求的算式则显示“Error!”。

把程序运行时测试用的算式文件“Comp.txt”（见附件及文件内容）保存到程序 P322.c 所在的文件夹且文件名保持不变。

可用素材: printf("Line %03d: Error!\n"...
printf("Line %03d: %5d + %-5d = %-6d\n"...
printf("Line %03d: %5d - %-5d = %-6d\n"...

提示: 建议使用 fgets() 读入一行到字符串、再使用 sscanf() 从字符串中读，如此逐行处理。程序的运行效果应类似地如图 322.1 所示。

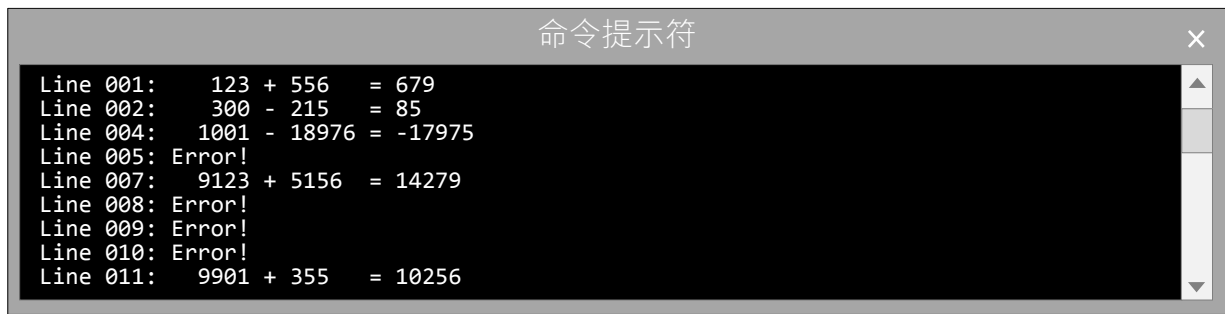


图 322.1

附件:  Comp.txt

文件内容:

Comp.txt

```
123 + 556
300 - 215

1001 - 18976
1001 * 5

9123 + 5156
808
    xym
200 +ab 100
9901  + 355
```

参考答案:

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

```
int main(void)
{
    FILE *fp;

    int num1, num2, i = 1;
    char op, str[201];

    fp = fopen("Comp.txt", "r");
    if (fp == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    while (!feof(fp))
    {
        if (fgets(str, 201, fp) != NULL)
        {
            if (sscanf(str, "%d %c %d", &num1, &op, &num2) == 3)
            {
                if (op == '+')
                {
                    printf("Line %03d: %5d + %-5d = %-6d\n", i, num1, num2, num1 + num2);
                }
                else if (op == '-')
                {
                    printf("Line %03d: %5d - %-5d = %-6d\n", i, num1, num2, num1 - num2);
                }
                else
                {

```

```

        printf("Line %03d: Error!\n", i);
    }
}
else if (sscanf(str, "%d %c %d", &num1, &op, &num2) != EOF)
{
    printf("Line %03d: Error!\n", i);
}
i++;
}
}

fclose(fp);

return 0;
}

```

P764

DIFFICUTLY ★★

从命令行输入源文件名（含路径）和目标文件名（含路径），实现将源文件复制到目标文件。

注意事项：

1. 源文件可能是文件文件，也可能是二进制文件。
2. 源文件名和目标文件名只从命令行输入，当命令行格式不正确（参数个数不为 3）时，应报错。
3. 程序的返回值（即由 main() 函数 return 的值和程序使用 exit() 终止运行时返回的值，也称退出代码）规定为：
 - a) 复制成功，返回 0。
 - b) 命令行格式不对，返回 1。
 - c) 源文件打开失败，返回 2。
 - d) 目标文件创建失败，返回 3
 - e) 向目标文件写数据的过程中出错，返回 4。
4. 向目标文件写数据的过程中出错的情况很少发生，用户根据图例中的输入数据进行测试时，很可能不会出错，但程序应考虑出错的情况（例如磁盘空间不够、往 U 盘上写一个大文件的过程中 U 盘出错或被拔走）。

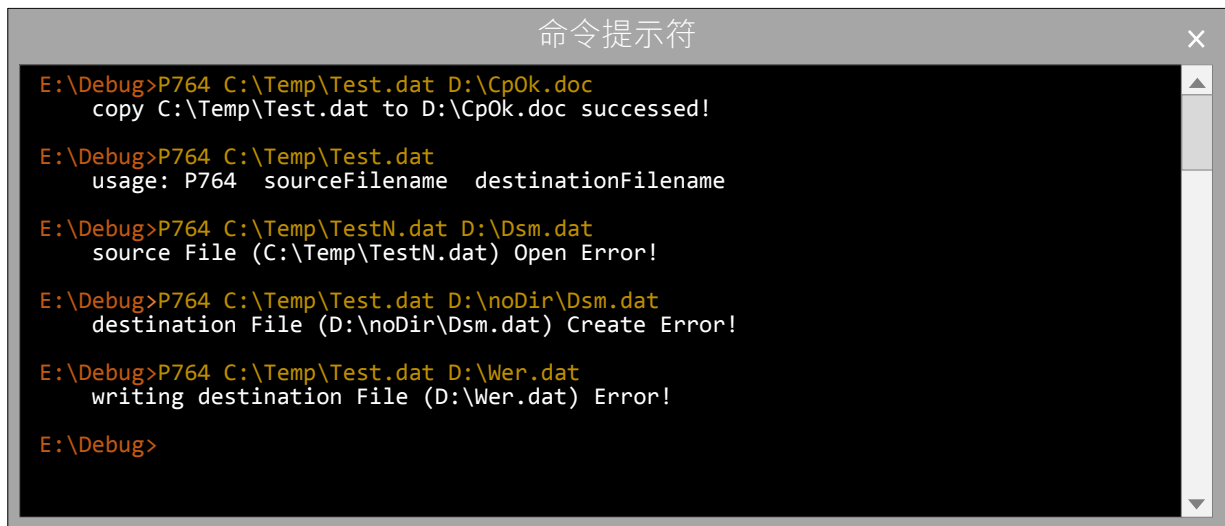
可用素材：

```

printf("    copy %s to %s succeeded!\n"...
printf("    usage: P764 sourceFilename destinationFilename\n")
printf("    source File (%s) Open Error!\n"...
printf("    destination File (%s) Create Error!\n"...
printf("    writing destination File (%s) Error!\n"...

```

程序的运行效果应类似地如图 764.1 所示，**橙色部分**为系统命令行提示符，表示程序 P764.exe 所在的文件夹，用户的程序位置可不必如此；**金色部分**是从命令行输入的内容。



```
E:\Debug>P764 C:\Temp\Test.dat D:\Cp0k.doc
copy C:\Temp\Test.dat to D:\Cp0k.doc succeeded!

E:\Debug>P764 C:\Temp\Test.dat
usage: P764 sourceFilename destinationFilename

E:\Debug>P764 C:\Temp\TestN.dat D:\Dsm.dat
source File (C:\Temp\TestN.dat) Open Error!

E:\Debug>P764 C:\Temp\Test.dat D:\noDir\Dsm.dat
destination File (D:\noDir\Dsm.dat) Create Error!

E:\Debug>P764 C:\Temp\Test.dat D:\Wer.dat
writing destination File (D:\Wer.dat) Error!

E:\Debug>
```

图 764.1

参考答案:

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

```
int main(int argc, char *argv[])
{
    FILE *fp1, *fp2;
    int ch;

    if (argc != 3)
    {
        printf("    usage: P764 sourceFilename destinationFilename\n");
        return 1;
    }

    fp1 = fopen(argv[1], "rb");
    if (fp1 == NULL)
    {
        printf("    source File (%s) Open Error!\n", argv[1]);
        return 2;
    }

    fp2 = fopen(argv[2], "wb");
    if (fp2 == NULL)
    {
        printf("    destination File (%s) Create Error!\n", argv[2]);
        fclose(fp1);
        return 3;
    }

    while (1)
    {
        ch = fgetc(fp1);
        if (ch != EOF)
        {
            if (fputc(ch, fp2) == EOF)
            {
                printf("    writing destination File (%s) Error!\n", argv[2]);
                fclose(fp1);
                fclose(fp2);
                return 4;
            }
        }
    }
    else
```

```

        {
            break;
        }
    }
    printf("    copy %s to %s succeeded!\n", argv[1], argv[2]);
    fclose(fp1);
    fclose(fp2);
    return 0;
}

```

P802

DIFFICUTLY ★★

程序 P802.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：有一存储很多商品数据（每件商品的属性先后包括：品名、规格、单价（有小数位）、数量，数据的最长长度分别为 20、10、6、5，在文件中以空格为分隔，每个商品的数据占一行）的文本文件，从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔，单价显示时只显示 2 位小数），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp.txt”（见附件及文件内容）保存到文件夹“D:\cExam”下且文件名保持不变。

可用素材： printf("\nmei you shang pin :...

程序的运行效果应类似地如图 802.1 和图 802.2 所示，金色部分是从键盘输入的内容。

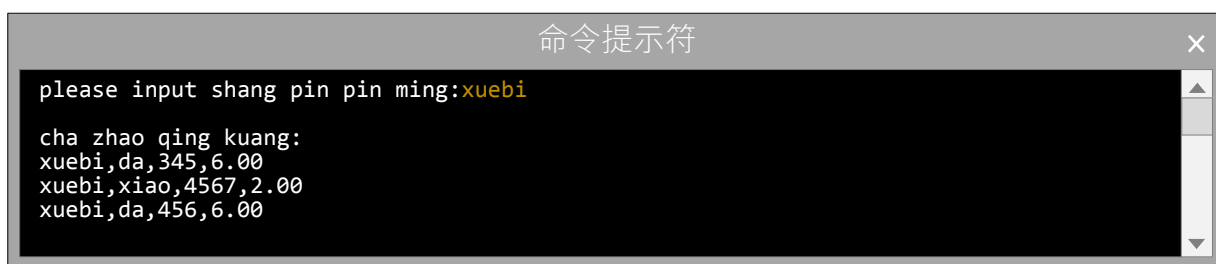


图 802.1 输入“xuebi”时



图 802.2 输入“kele”时

附件：  
P802.c sp.txt

文件内容：

P802.c

```

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

```

/* User Code Begin(考生可在本行后添加代码，定义程序中使用的结构体类型，行数不限) */

<A>

/* User Code End(考生添加代码结束) */

int main(void)

{

int n;

char sppm[21];

FILE *fp;

struct goods SP;

printf("Please input shang pin pin ming:");

gets(sppm);

fp = fopen("d:\\cExam\\sp.txt", "r");

if (NULL == fp)

{

printf("file open error!");

exit(0);

}

n = 0;

printf("\ncha zhao qing kuang:");

/* User Code Begin(考生可在本行后添加代码, 行数不限) */

/* User Code End(考生添加代码结束) */

if (strcmp(sppm, SP.pinming) == 0)

{

n++;

printf("\n%s,%s,%ld,%.2f", SP.pinming, SP.guige, SP.shuliang, SP.danjia);

}

/* User Code Begin(考生可在本行后添加代码, 行数不限) */

<C>

/* User Code End(考生添加代码结束) */

return 0;

}

sp.txt

xuebi da 6.00 345

nongfuSQxianchengduo zhongxingA 4.392 57398

xuebi xiao 2.004 4567

xuebi da 6.003 456

参考答案:

<A> struct goods

{

char pinming[21], guige[11];

float danjia;

long shuliang;

};

 while (!feof(fp))

{

if (fscanf(fp, "%s%s%f%d", SP.pinming, SP.guige, &SP.danjia, &SP.shuliang) == 4)

{

}

<C>

```

}
if (n == 0)
{
    printf("\nmei you shang pin :%s\n", sppm);
}

fclose(fp);

```

P805

DIFFICUTLY ★★

程序 P805.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：有一存储很多商品数据（每件商品的属性先后包括：品名、规格、数量、单价，编程时相应的数据类型分别定义为字符串 char(20)、字符串 char(12)、long、float）的二进制文件“sp.dat”（即未作任何格式转换而直接使用 fwrite() 将商品属性写入文件），从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp.dat”（见附件）保存到文件夹“D:\cExam”下且文件名保持不变。

可用素材： printf("mei you shang pin :...

程序的运行效果应类似地如图 805.1 和图 805.2 所示，金色部分是从键盘输入的内容。

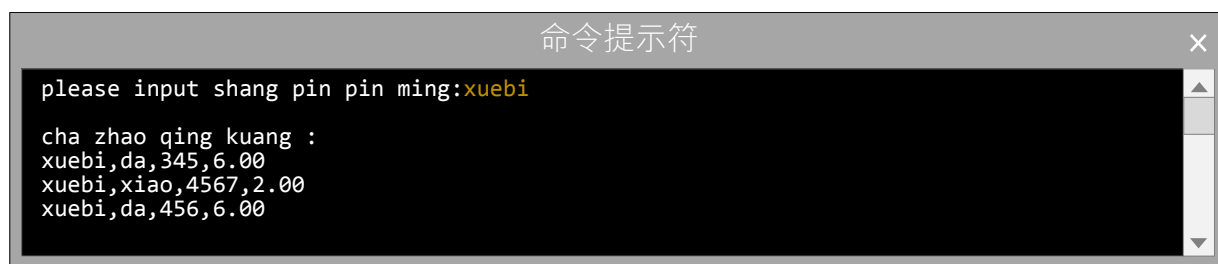


图 805.1 输入“xuebi”时



图 805.2 输入“kele”时

附件：  
P805.c sp.dat

文件内容：

P805.c

```

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

```

/* User Code Begin(考生可在本行后添加代码，定义程序中使用的结构体类型，行数不限) */

<A>

```

/* User Code End(考生添加代码结束) */

int main(void)
{
    int n;
    char sppm[20];
    struct goods SP;
    FILE *fp;

    printf("Please input shang pin pin ming:");
    gets(sppm);

    fp = fopen("d:\\Exam\\sp.dat", "rb");
    if (NULL == fp)
    {
        printf("file open error!");
        exit(0);
    }

    n = 0;
    printf("\ncha zhao qing kuang :");
    /* User Code Begin(考生可在本行后添加代码, 行数不限) */
    <B>
    /* User Code End(考生添加代码结束) */
    if (strcmp(sppm, SP.pinming) == 0)
    {
        n++;
        printf("\n%s,%s,%ld,%.2f", SP.pinming, SP.guige, SP.shuliang, SP.danjia);
    }
    /* User Code Begin(考生可在本行后添加代码, 行数不限) */
    <C>
    /* User Code End(考生添加代码结束) */

    return 0;
}

```

测试内容备选: xuebi、xianchengduo

参考答案:

```

<A> struct goods
{
    char pinming[20], guige[12];
    long shuliang;
    float danjia;
};
<B> while (!feof(fp))
{
    if (fread(&SP, sizeof(struct goods), 1, fp) == 1)
    {
        <C>
    }
    if (n == 0)
    {
        printf("\nmei you shang pin :%s\n", sppm);
    }

    fclose(fp);
}

```

P809

DIFFICUTLY ★★

程序 P809.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：有一存储很多商品数据（每件商品的属性先后包括：品名、规格、单价（有小数位）、数量，数据的最长长度分别为 20、10、6、5，在文件中以空格为分隔，每个商品的数据占一行）的文本文件，从键盘输入某种商品的品名，要求在文件中查找有无相应品名商品（可能有多条记录或没有），若有则在屏幕上显示出相应的商品的品名、规格、数量、单价（显示时，品名、规格、数量、单价之间使用逗号“,”作分隔，单价显示时只显示 2 位小数），若无则显示没有相应品名的商品。

把程序运行时测试用的商品数据文件“sp.txt”（见附件及文件内容）保存到程序 P809.c 所在的文件夹且文件名保持不变

可用素材： `printf("\nmei you shang pin :...`

程序的运行效果应类似地如图 809.1 和图 809.2 所示，金色部分是从键盘输入的内容。

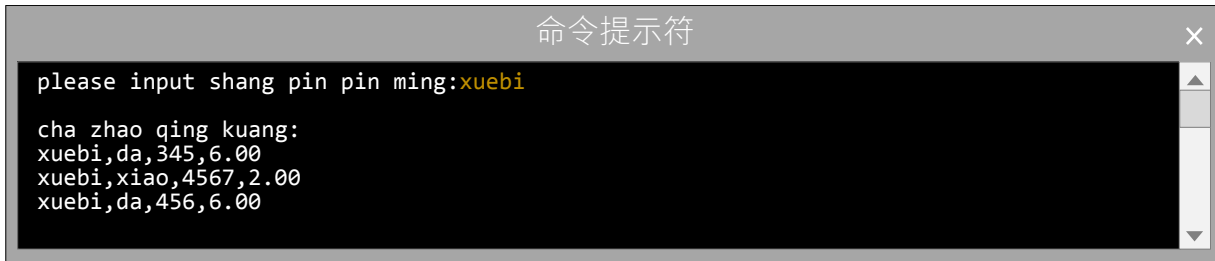


图 809.1 输入“xuebi”时



图 809.2 输入“kele”时

附件：  
P809.c sp.txt

文件内容：

P809.c

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

struct goods
{
    char pinming[21];
    char guige[11];
    long int shuliang;
    float danjia;
};

int main(void)
{
    int n;
    char sppm[21];
```

```

FILE *fp;
struct goods SP;

printf("Please input shang pin pin ming:");
gets(sppm);

/* User Code Begin(考生可在本行后添加代码, 行数不限) */
<A>
/* User Code End(考生添加代码结束) */

n = 0;
printf("\ncha zhao qing kuang:");
/* User Code Begin(考生可在本行后添加代码, 行数不限) */
<B>
/* User Code End(考生添加代码结束) */
    if (strcmp(sppm, SP.pinming) == 0)
    {
        n++;
        printf("\n%s,%s,%ld,%.2f", SP.pinming, SP.guige, SP.shuliang, SP.danjia);
    }
/* User Code Begin(考生可在本行后添加代码, 行数不限) */
<C>
/* User Code End(考生添加代码结束) */

return 0;
}

```

sp.txt

```

xuebi da 6.00 345
nongfuSQxianchengduo zhongxingA 4.392 57398
xuebi xiao 2.004 4567
xuebi da 6.003 456

```

参考答案:

```

<A> fp = fopen("sp.txt", "r");
    if (NULL == fp)
    {
        printf("file open error!");
        exit(0);
    }
<B> while (!feof(fp))
    {
        if (fscanf(fp, "%s%s%f%d", SP.pinming, SP.guige, &SP.danjia, &SP.shuliang) == 4)
        {
<C>
        }
    }
    if (n == 0)
    {
        printf("\nmei you shang pin :%s\n", sppm);
    }

    fclose(fp);

```

P313


DIFFICUTLY ★★★

求任意的一个 $m \times n$ 矩阵的鞍点——鞍点是指该位置上的元素在该行上为最大、在该列上为最小，矩阵中可能没有鞍点，但最多只有一个鞍点。 m 、 n ($2 \leq m \leq 20$ 、 $2 \leq n \leq 20$) 及矩阵元素从键盘输入（只考虑 int 型和每行、每列中没有并列最大/最小的情况）。

可用素材:

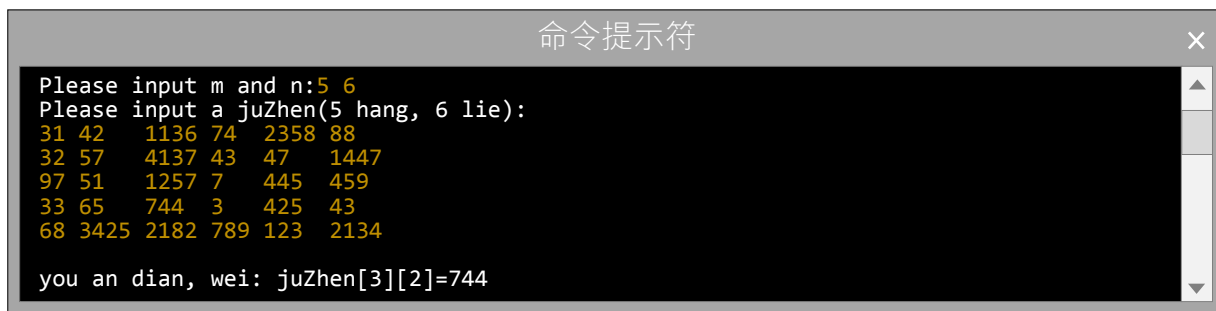
```
printf("Please input m and n:");
printf("Please input a juZhen(... hang, ... lie):\n...");
printf("\nmei you an dian.\n");
printf("\nyou an dian, wei: juZhen[...][...]=...\n...");
```

程序的运行效果应类似地如图 313.1 和图 313.2 所示，金色部分是从键盘输入的内容。



```
命令提示符
Please input m and n:5 6
Please input a juZhen(5 hang, 6 lie):
31 42 36 74 2358 88
32 57 37 43 47 1447
97 51 257 7 445 459
33 65 44 3 425 43
68 3425 82 789 123 2134
mei you an dian.
```

图 313.1 无鞍点



```
命令提示符
Please input m and n:5 6
Please input a juZhen(5 hang, 6 lie):
31 42 1136 74 2358 88
32 57 4137 43 47 1447
97 51 1257 7 445 459
33 65 744 3 425 43
68 3425 2182 789 123 2134
you an dian, wei: juZhen[3][2]=744
```

图 313.2 有鞍点

参考答案:

```
#include <stdio.h>

int main(void)
{
    int m0, n0, arr[20][20], i, j, k, max, min, maxi, maxj, mini, minj, flag = 0;

    do
    {
        printf("Please input m and n:");
        scanf("%d %d", &m0, &n0);
    } while (m0 < 2 || m0 > 20 || n0 < 2 || n0 > 20);

    printf("Please input a juZhen(%d hang, %d lie):\n", m0, n0);
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < n0; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }

    for (i = 0; i < m0; i++)
```

```

{
    max = arr[i][0];
    maxi = i;
    maxj = 0;

    for (j = 0; j < n0; j++)
    {
        if (arr[i][j] > max)
        {
            max = arr[i][j];
            maxi = i;
            maxj = j;
        }
    }

    min = arr[0][maxj];
    mini = 0;
    minj = maxj;

    for (k = 0; k < m0; k++)
    {
        if (arr[k][maxj] < min)
        {
            min = arr[k][maxj];
            mini = k;
            minj = maxj;
        }
    }

    if (maxi == mini && maxj == minj)
    {
        printf("\nyou an dian, wei: juZhen[%d][%d]=%d\n", maxi, maxj, arr[maxi][maxj]);
        flag = 1;
        break;
    }
}
if (flag == 0)
{
    printf("\nmei you an dian.\n");
}

return 0;
}

```

P329

DIFFICUTLY ★★★

有若干（最多 50 人）学生的信息（学号、姓名、性别、出生年、C 语言、英语、微积分）存储在名为“学生成绩.txt”的文本文件中，信息的存储格式为信息标题占第 1 行，其余每行为一学生的信息，每一学生的信息项之间以 1 个或多个 TAB（制表符）作为分隔，假定每一学生的信息均是完整的，不存在错误。

把程序运行时测试用的文件“学生成绩.txt”（见附件及文件内容）保存到程序 P329.c 所在的文件夹且文件名保持不变。

编程计算这些学生的平均成绩，并以平均成绩从高到低、按图 329.1 所示的格式在屏幕上输出这些学生的相关信息。

可用素材： puts("学生成绩文件“学生成绩.txt”打开失败，请仔细检查文件名是否正确，对应文件是否存在！")
 puts("名次 平均成绩 学号 姓名 性别 出生年 C语言 英语 微积分")

```
printf("%3d %7.2f %d %-8s %-2s %d %3d %3d %3d\n"...
```

程序的运行效果应类似地如图 329.1 所示。

命令提示符								
名次	平均成绩	学号	姓名	性别	出生年	C语言	英语	微积分
1	88.33	2008122625	屈哲律	男	1990	91	79	95
2	84.33	2008122624	崔艺丹	女	1989	83	87	83
3	75.33	2008122628	戚桂兰	女	1990	74	79	73
4	74.67	2008122629	李秋丽	女	1991	71	90	63
5	71.67	2008122626	陈筑夏	女	1991	71	75	69
6	49.00	2008122627	陈伟	男	1989	66	51	30

图 329.1

附件:  学生成绩.txt

文件内容:

学生成绩.txt

学号	姓名	性别	出生年	C语言	英语	微积分
2008122624	崔艺丹	女	1989	83	87	83
2008122625	屈哲律	男	1990	91	79	95
2008122626	陈筑夏	女	1991	71	75	69
2008122627	陈伟	男	1989	66	51	30
2008122628	戚桂兰	女	1990	74	79	73
2008122629	李秋丽	女	1991	71	90	63

参考答案:

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

```
struct student
```

```
{
    int num;
    char name[10], sex[10];
    int birth, Clan, Eng, cal;
    float avg;
};
```

```
int main(void)
```

```
{
    FILE *fp;
    struct student ss[50], tmp;
    char ch;
    int i = 0, j, count = 0;

    fp = fopen("学生成绩.txt", "r");
    if (NULL == fp)
    {
        puts("学生成绩文件“学生成绩.txt”打开失败, 请仔细检查文件名是否正确, 对应文件是否存在!");
        return 0;
    }
}
```

```
while (1)
```

```
{
    ch = fgetc(fp);
    if (ch == '\n')
    {
        break;
    }
}
```

```
while (fscanf(fp, "%d%s%d%d%d", &ss[i].num, ss[i].name, ss[i].sex, &ss[i].birth,
```

```

        &ss[i].Clan, &ss[i].Eng, &ss[i].cal) == 7)
    {
        count++;
        ss[i].avg = (ss[i].Clan + ss[i].Eng + ss[i].cal) / (float)3;
        i++;
    }

    for (j = 1; j < count; j++)
    {
        for (i = 0; i < count - j; i++)
        {
            if (ss[i].avg < ss[i + 1].avg)
            {
                tmp = ss[i];
                ss[i] = ss[i + 1];
                ss[i + 1] = tmp;
            }
        }
    }

    puts("名次 平均成绩 学号      姓名      性别  出生年  C语言  英语  微积分");
    for (i = 0; i < count; i++)
    {
        printf("%3d  %7.2f  %d  %-8s  %-2s  %d  %3d  %3d  %3d\n",
            i + 1, ss[i].avg, ss[i].num, ss[i].name, ss[i].sex,
            ss[i].birth, ss[i].Clan, ss[i].Eng, ss[i].cal);
    }

    fclose(fp);

    return 0;
}

```

P412

DIFFICUTLY ★★★

编写一程序实现以下功能:

1. 程序运行时先显示“Please input numbers:”，再从键盘上读入一组整数（只考虑 int 型），数与数之间只使用空格或回车作分隔。数可正可负，最多 10000 个，但若读入的数为 -222 时，则表示输入结束且 -222 不算在该组数内。
2. 对这一组数按从小到大的顺序进行排序。
3. 将排序后的这一组数输出到屏幕上，输出格式为每行 6 个数，数与数之间使用逗号“,”分隔，两个逗号之间的宽度（不算逗号）为 6 且使用左对齐格式。注意，行尾没有逗号。

可用素材: `printf("Please input numbers: ")`
`printf("\nOutput:\n")`

程序的运行效果应类似地如图 412.1 所示，金色部分是从键盘输入的内容。

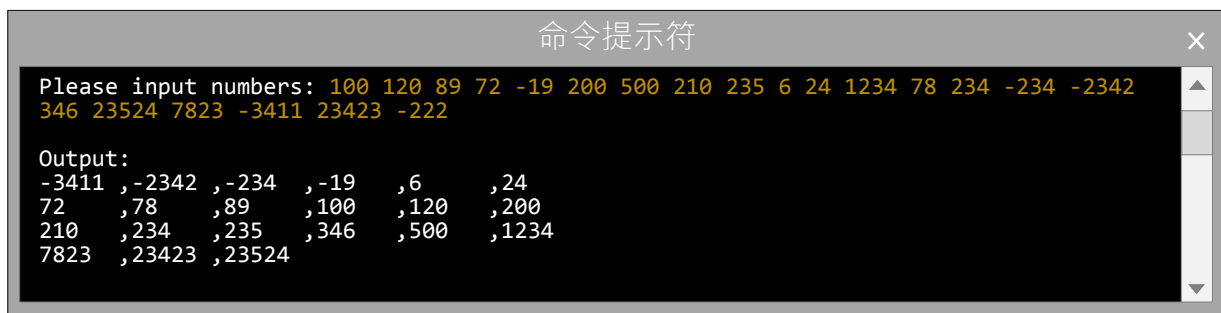


图 412.1

参考答案:

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

```
int main(void)
{
    int arr[10000], tmp, i, j, count = 0;

    printf("Please input numbers: ");
    for (i = 0; i < 10000; i++)
    {
        scanf("%d", &tmp);
        if (tmp != -222)
        {
            arr[i] = tmp;
            count++;
        }
        else
        {
            break;
        }
    }

    for (j = 1; j < count; j++)
    {
        for (i = 0; i < count - j; i++)
        {
            if (arr[i] > arr[i + 1])
            {
                tmp = arr[i];
                arr[i] = arr[i + 1];
                arr[i + 1] = tmp;
            }
        }
    }

    printf("\nOutput:\n");
    for (i = 0; i < count; i++)
    {
        if ((i + 1) % 6 == 0 || i == count - 1)
        {
            printf("%-6d\n", arr[i]);
        }
        else
        {
            printf("%-6d,", arr[i]);
        }
    }

    return 0;
}
```


P415

DIFFICUTLY ★★★

设有 10 名歌手（编号为 1 ~ 10）参加歌咏比赛，另有 6 名评委打分，每位歌手的得分从键盘输入：先提示 “Please input singer's score: ”，再依次输入第 1 个歌手的 6 位评委打分（10 分制，分数为整型，分数之间使用空格分隔），第 2 个歌手的 6 位评委打分……以此类推。计算出每位歌手的最终得分（扣除一个最高分和一个最低分后的平均分，最终得分保留 2 位小数），最后按最终得分由高到低的顺序输出每位歌手的编号及最终得分。

注：变量数据类型的选择应适当，在保证满足设计要求精度的情况下，养成不浪费内存空间和计算时间的好习惯。

可用素材： `printf("Please input singer's score:\n")`
`printf("\nscores:\n")`
`printf("No....: ...`

程序的运行效果应类似地如图 415.1 所示，金色部分是从键盘输入的内容。



图 415.1

参考答案：

```
#include <stdio.h>

struct data
{
    int num, score[6];
    float final;
};

void GetAvg(struct data singer[], int num);
void Sort(struct data singer[], int num);

int main(void)
{
    struct data singer[10], *psing = singer;
    int i, j;

    printf("Please input singer's score:\n");
```

```

    for (i = 0; i < 10; i++)
    {
        singer[i].num = i + 1;
        for (j = 0; j < 6; j++)
        {
            scanf("%d", &singer[i].score[j]);
        }
    }
    GetAvg(singer, 10);

    printf("\nscores:\n");
    Sort(singer, 10);
    for (i = 0; i < 10; i++)
    {
        printf("No.%-2d: %.2f\n", (psing + i)->num, (psing + i)->final);
    }

    return 0;
}

void GetAvg(struct data singer[], int num)
{
    int i, j, max, min, sum;

    for (i = 0; i < num; i++)
    {
        max = min = singer[i].score[0];
        sum = singer[i].score[0];
        for (j = 1; j < 6; j++)
        {
            if (singer[i].score[j] > max)
            {
                max = singer[i].score[j];
            }
            if (singer[i].score[j] < min)
            {
                min = singer[i].score[j];
            }
            sum += singer[i].score[j];
        }
        sum = sum - max - min;
        singer[i].final = sum / (float)4;
    }
}

void Sort(struct data singer[], int num)
{
    struct data tmp;
    int i, j;

    for (j = 1; j < num; j++)
    {
        for (i = 0; i < num - j; i++)
        {
            if (singer[i].final < singer[i + 1].final)
            {
                tmp = singer[i];
                singer[i] = singer[i + 1];
                singer[i + 1] = tmp;
            }
        }
    }
}

```

P419

DIFFICUTLY ★★★

编写一程序实现以下功能:

1. 程序运行时先显示 “Input:”，再从键盘上读入一组整数（只考虑 int 型），数与数之间只使用空格或回车作分隔。数可正可负，最多 100 个，但若读入的数为 -888 时，则表示输入结束且 -888 不算在该组数内。
2. 对这一组数按从小到大的顺序进行排序。
3. 将排序后的这一组数输出到屏幕上。

可用素材: `printf("Input: ")`
`printf("\nResult: ")`
`printf("%d ...`

程序的运行效果应类似地如图 419.1 所示，金色部分是从键盘输入的内容。

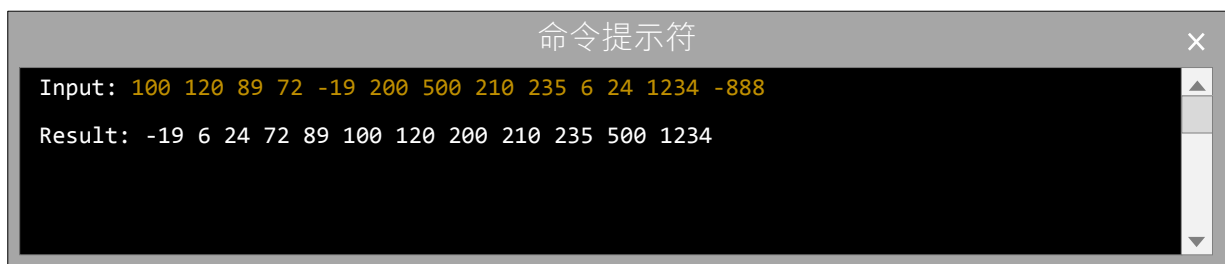


图 419.1

参考答案:

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

```
int main(void)
{
    int arr[100], tmp, i, j, count = 0;

    printf("Input: ");
    for (i = 0; i < 100; i++)
    {
        scanf("%d", &tmp);
        if (tmp != -888)
        {
            arr[count] = tmp;
            count++;
        }
        else
        {
            break;
        }
    }

    for (j = 1; j < count; j++)
    {
        for (i = 0; i < count - j; i++)
        {
            if (arr[i] > arr[i + 1])
            {
                tmp = arr[i];
                arr[i] = arr[i + 1];
                arr[i + 1] = tmp;
            }
        }
    }
}
```

```

    }

    printf("\nResult: ");
    for (i = 0; i < count; i++)
    {
        printf("%d ", arr[i]);
    }
    putchar('\n');

    return 0;
}

```

P719

DIFFICUTLY ★★★

从键盘上输入 5 个字符串（约定：每个字符串中字符数 ≤ 80 字节），对其进行升序排序并输出。

可用素材： `printf("Input 5 strings:\n")`
`printf("-----\n")`

程序的运行效果应类似地如图 719.1 所示，金色部分是从键盘输入的内容。



图 719.1

参考答案：

```

#include <stdio.h>
#include <string.h>

int main(void)
{
    char str[5][81], tmp[81];
    int i, j;

    printf("Input 5 strings:\n");
    for (i = 0; i < 5; i++)
    {
        gets(str[i]);
    }

    for (j = 1; j < 5; j++)
    {
        for (i = 0; i < 5 - j; i++)
        {
            if (strcmp(str[i], str[i + 1]) > 0)
            {
                strcpy(tmp, str[i]);
                strcpy(str[i], str[i + 1]);
                strcpy(str[i + 1], tmp);
            }
        }
    }
}

```

```

    }
}
printf("-----\n");
for (i = 0; i < 5; i++)
{
    puts(str[i]);
}

return 0;
}

```

P765

DIFFICUTLY ★★★

程序 P765.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：有五个学生，每个学生的数据包括学号、姓名（最长 19 字节）、三门课的成绩，从键盘输入五个学生的数据，计算每个学生的平均成绩并按平均成绩由高到低排序，并将排序结果显示。

注：要求用结构体编程，变量数据类型的选择应适当，在保证满足设计要求精度的情况下，养成不浪费内存空间和计算时间的好习惯。

可用素材： `printf("Please input info of students:No Name Math English Computer\n")`

程序的运行效果应类似地如图 765.1 所示，金色部分是从键盘输入的内容。

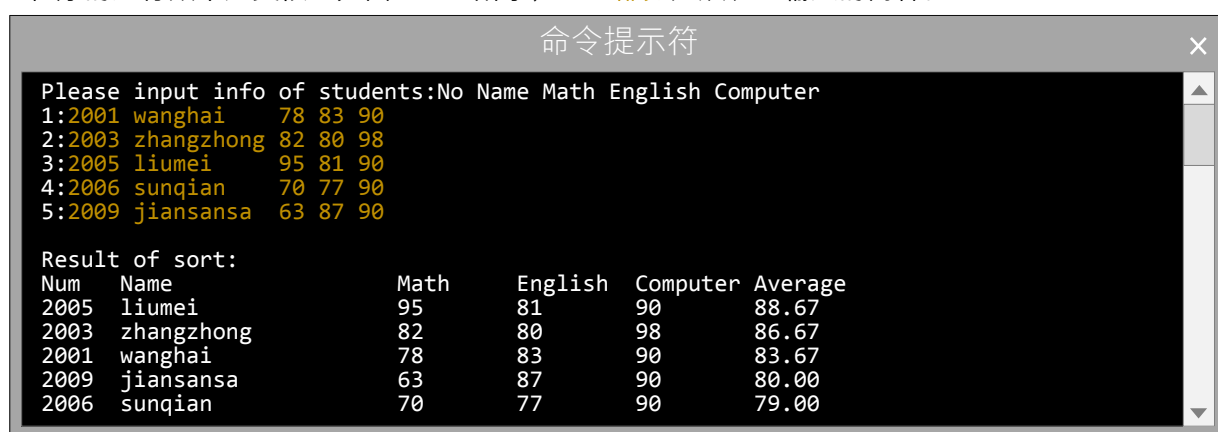


图 765.1

附件:  P765.c

文件内容:

P765.c

```

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

/* User Code Begin(考生可在本行后添加代码，行数不限) */
<A>
/* User Code End(考生添加代码结束) */

int main(void)
{
    STUDENT myclass[5], *pStu = myclass;
    int i;
    const int N = 5;

```

```

/* User Code Begin(考生可在本行后添加代码, 行数不限) */
<B>
[Redacted Code]

/* User Code End(考生添加代码结束) */

printf("\nResult of sort:\n");
printf("Num    Name                Math    English    Computer    Average\n");
for (i=0; i<N; i++)
{
    printf("%-5d %-20s %-8d %-8d %-8d %-.2f\n", (pStu+i)->num, (pStu+i)->name,
        (pStu+i)->math, (pStu+i)->english, (pStu+i)->computer, (pStu+i)->average);
}

return 0;
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分, 行数不限) */
<C>
[Redacted Code]

```

参考答案:

```

<A> typedef struct student
{
    int num;
    char name[20];
    int math, english, computer;
    float average;
} STUDENT;

void Sort(STUDENT myclass[], int num);
<B> printf("Please input info of students:No Name Math English Computer\n");
for (i = 0; i < 5; i++)
{
    printf("%d:", i + 1);
    scanf("%d%s%d%d%d", &myclass[i].num, myclass[i].name,
        &myclass[i].math, &myclass[i].english, &myclass[i].computer);
    myclass[i].average = (myclass[i].math + myclass[i].english + myclass[i].computer)
        / (float)3;
}
Sort(myclass, 5);
<C> void Sort(STUDENT myclass[], int num)
{
    int i, j;
    STUDENT tmp;

    for (j = 1; j < num; j++)
    {
        for (i = 0; i < num - j; i++)
        {
            if (myclass[i].average < myclass[i + 1].average)
            {
                tmp = myclass[i];
                myclass[i] = myclass[i + 1];
                myclass[i + 1] = tmp;
            }
        }
    }
}

```

P767

DIFFICUTLY ★★★

从键盘读入一行字符（约定：字符数 ≤ 127 字节，其中的空格不固定、有多有少）和加密间隔（假定只输入正整数），将加密后的文字输出。

具体加密方法示例为：对于输入“1 2 3 4 5 6 7 8 9abcd ef ghiA BCD EFG HI XYZ”、加密间隔为 9，先去掉输入中的空格后输出为：123456789abcdefghijklmABCDEFGHIXYZ，然后按间隔 9 分组：

```
123456789
abcdefghijklm
ABCDEFGHI
XYZ
```

输出密码的方法为：从第 1 组开始，依次从每组各取 1 字符输出，每一轮取完后输出一个空格，则以上输入的密码输出为：1aAX 2bBY 3cCZ 4dD 5eE 6fF 7gG 8hH 9iI。

可用素材：

```
printf("Input a string:")
printf("\nInput jiange:")
printf("\nThe string of deleted space:")
printf("\nThe result is:")
```

程序的运行效果应类似地如图 767.1 所示，金色部分是从键盘输入的内容。



图 767.1

参考答案：

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

int main(void)
{
    char str1[128], str2[128], result[127][128];
    int num, i, j, k, len, div, count = 0;

    printf("Input a string:");
    gets(str1);
    printf("\nInput jiange:");
    scanf("%d", &num);

    for (i = 0, j = 0; str1[i] != '\0'; i++)
    {
        if (str1[i] != ' ')
        {
            str2[j] = str1[i];
            j++;
        }
    }
    str2[j] = '\0';
    printf("\nThe string of deleted space:");
    puts(str2);
```

```

for (i = 0, j = 0, k = 0; str2[i] != '\0'; i++)
{
    result[j][k] = str2[i];
    k++;
    count++;
    if (count == num)
    {
        result[j][k] = '\0';
        j++;
        k = 0;
        count = 0;
    }
}

len = strlen(str2);
div = len / num;
printf("\nThe result is:");
for (j = 0, k = 0; j < num; j++)
{
    for (i = 0; i < div; i++)
    {
        printf("%c", result[i][j]);
    }
    if (k < len % num)
    {
        printf("%c", result[div][k]);
        k++;
    }
    putchar(' ');
}
putchar('\n');

return 0;
}

```

P769

DIFFICUTLY ★★★

输出 m 和 n 之间的回文素数， m 和 n 从键盘读入（假定满足 $5 \leq m \leq n \leq 100000$ ），回文是指正向与反向的字符都一样，例如 1、11、101、131 等。

可用素材： `printf("please input m, n(5<=m<=n<=100000): ")`
`printf("Result(%d-%d):\n"...`
`printf("%d "...`

程序的运行效果应类似地如图 769.1 所示，金色部分是从键盘输入的内容。



图 769.1

参考答案：

```

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

```



```

int main(void)
{
    int m0, n0, tmp, i, j, k, arr[100000], count = 0, flag, len;
    char str[7];

    do
    {
        printf("please input m, n(5<=m<=n<=100000): ");
        scanf("%d,%d", &m0, &n0);
    } while (m0 < 5 || m0 > 100000 || n0 < 5 || n0 > 100000);

    if (m0 > n0)
    {
        tmp = m0;
        m0 = n0;
        n0 = tmp;
    }

    for (i = m0; i <= n0; i++)
    {
        flag = 0;
        for (j = 2; j < i; j++)
        {
            if (i % j == 0)
            {
                flag = 1;
                break;
            }
        }
        if (flag == 0)
        {
            itoa(i, str, 10);
            len = strlen(str);
            for (j = 0, k = len - 1; j < k; j++, k--)
            {
                if (str[j] != str[k])
                {
                    flag = 1;
                    break;
                }
            }
        }
        if (flag == 0)
        {
            arr[count] = i;
            count++;
        }
    }

    printf("Result(%d-%d):\n", m0, n0);
    for (i = 0; i < count; i++)
    {
        printf("%d ", arr[i]);
    }
    putchar('\n');

    return 0;
}

```

P772

DIFFICUTLY ★★★

对从键盘输入的一行字符（约定：字符数 ≤ 127 字节）进行排序（按每个字符的 ASCII 码由小到大）并输出。

注：程序中不能使用库函数 `gets()`、`fgets()` 或使用同名的变量、函数、单词。

可用素材： `printf("input the string: ")`
`printf("\nResult: ")`

程序的运行效果应类似地如图 772.1 所示，金色部分是从键盘输入的内容。



图 772.1

参考答案：

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

int main(void)
{
    char str[128], ch, tmp;
    int len, i, j;

    printf("input the string: ");
    for (i = 0; i < 128; i++)
    {
        ch = getchar();
        if (ch != '\n')
        {
            str[i] = ch;
        }
        else
        {
            break;
        }
    }
    str[i] = '\0'; /* 另解：一行字符的输入可仅使用一条语句 scanf("%[^\\n]", str) 实现 */
    len = strlen(str);

    for (j = 1; j < len; j++)
    {
        for (i = 0; i < len - j; i++)
        {
            if (str[i] > str[i + 1])
            {
                tmp = str[i];
                str[i] = str[i + 1];
                str[i + 1] = tmp;
            }
        }
    }

    printf("\nResult: %s\n", str);
}
```

```
    return 0;
}
```

P815

DIFFICUTLY ★★★

程序 P815.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `<A>` 换成代码）。


程序的功能是：从键盘上输入 5 个字符串（约定：每个字符串中字符数 ≤ 80 字节），对其进行升序排序并输出。

可用素材： `printf("Input 5 strings:\n")`
`printf("-----\n")`

程序的运行效果应类似地如图 815.1 所示，金色部分是从键盘输入的内容。



图 815.1

附件:  P815.c

文件内容:

P815.c

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

#define MAX_LINE 5
#define MAX_LINE_LEN 81

/* 本部分代码功能建议：函数原型声明 */
/* User Code Begin(Limit: lines<=1, lineLen<=50, 考生可在本行后添加代码、最多1行、每行长<=50字符) */
<A>
/* User Code End(考生添加代码结束。注意：空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

int main(void)
{
    int i;
    char *pstr[MAX_LINE], str[MAX_LINE][MAX_LINE_LEN];

    for (i=0; i<MAX_LINE; i++)
    {
        pstr[i] = str[i];
    }

    printf("Input 5 strings:\n");
    for (i=0; i<MAX_LINE; i++)
    {
```

```

    gets(pstr[i]);
}

sortP_Str(pstr);
printf("-----\n");
for (i=0; i<MAX_LINE; i++)
{
    printf("%s\n", pstr[i]);
}

return 0;
}

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */
<B>

```

参考答案:

```

<A> void sortP_Str(char *pstr[MAX_LINE_LEN]);
<B> void sortP_Str(char *pstr[MAX_LINE_LEN])
{
    int i, j;
    char tmp[MAX_LINE_LEN];

    for (j = 1; j < MAX_LINE; j++)
    {
        for (i = 0; i < MAX_LINE - j; i++)
        {
            if (strcmp(pstr[i], pstr[i + 1]) > 0)
            {
                strcpy(tmp, pstr[i]);
                strcpy(pstr[i], pstr[i + 1]);
                strcpy(pstr[i + 1], tmp);
            }
        }
    }
}

```

P315

DIFFICUTLY ★★★

程序运行时, 先从键盘输入一个文本文件的文件名(约定: 字符数 ≤ 127 字节, 可含路径)和一个字符串(约定: 字符数 ≤ 20 字节, 其中不含空格、TAB 等, 后面称之为 **Str**), 再在屏幕上显示该文件的内容。要求显示完内容后, 在屏幕上输出文件的行数(行之间以 '\n' 为分隔、每行的长度不定但均 ≤ 200 个字节)、字符串 **Str** 在文件中第 1 次出现的行号和最后一次出现的行号(查找时不区分大小写、不跨行查找, 若未找到, 则行号显示为 -1)。

把程序运行时测试用的文件“Test.txt”(见附件及文件内容)保存到计算机。

可用素材:

```

printf("input the file's name and the string: ")
printf("\nfile open error!")
printf("-----File content:-----\n")
printf("\n-----File summary:-----\n")
printf("... lines, first line: ..., last line: ...\n")


```

程序的运行效果应类似地如图 315.1 所示, 金色部分是从键盘输入的内容。



```
命令提示符
input the file's name and the string: C:\Temp\Test.txt Value
-----File content:-----
/*      stdlib.h
   Definitions for common types, variables, and functions.
   Copyright (c) Borland International 1987,1988
   All Rights Reserved.
*/
char  *_Cdecl ltoa      (long value, char *string, int radix);
int    _Cdecl putenv    (const char *name);
unsigned _Cdecl _rotl    (unsigned value, int count);
unsigned _Cdecl _rotr    (unsigned value, int count);
void    _Cdecl swab      (char *from, char *to, int nbytes);
char  *_Cdecl ultoa      (unsigned long kvAluE, char *string, int radix);
-----File summary:-----
11 lines, first line: 6, last line: 11
```

图 315.1

附件:  Test.txt

文件内容:

Test.txt

```
/*      stdlib.h
   Definitions for common types, variables, and functions.
   Copyright (c) Borland International 1987,1988
   All Rights Reserved.
*/
char  *_Cdecl ltoa      (long value, char *string, int radix);
int    _Cdecl putenv    (const char *name);
unsigned _Cdecl _rotl    (unsigned value, int count);
unsigned _Cdecl _rotr    (unsigned value, int count);
void    _Cdecl swab      (char *from, char *to, int nbytes);
char  *_Cdecl ultoa      (unsigned long kvAluE, char *string, int radix);
```

参考答案:

```
#include <stdio.h>

int main(void)
{
    FILE *fp;
    char str[128], search[21], line[201];
    int i = 0, j, k, l0, ch, linenum = 0, flag = 0, first = -1, last = -1;

    printf("input the file's name: ");
    scanf("%s%s", str, search);

    fp = fopen(str, "r");
    if (fp == NULL)
    {
        printf("File open error!\n");
        return 0;
    }

    printf("-----File content:-----\n");
    while (!feof(fp))
    {
        ch = fgetc(fp);
        line[i] = ch;
        i++;
    }
```

```

    if (ch == '\n')
    {
        linenum++;
        putchar(ch);
        line[i] = '\0';
        i = 0;
    }
    else if (ch == EOF)
    {
        linenum++;
        line[i] = '\0';
    }
    else
    {
        putchar(ch);
    }

    if (ch == '\n' || ch == EOF)
    {
        for (j = 0, k = 0; line[j] != '\0'; j++)
        {
            if (line[j] == search[k] || line[j] == search[k] - 32 ||
                line[j] == search[k] + 32)
            {
                l0 = j;
                while ((line[l0] == search[k] || line[l0] == search[k] - 32 ||
                    line[l0] == search[k] + 32) && search[k] != '\0')
                {
                    k++;
                    l0++;
                }
                if (search[k] == '\0')
                {
                    if (flag == 0)
                    {
                        first = linenum;
                        flag = 1;
                    }
                    last = linenum;
                }
                else
                {
                    k = 0;
                }
            }
        }
    }

    printf("\n-----File summary:-----\n");
    printf("%d lines, first line: %d, last line: %d\n", linenum, first, last);

    fclose(fp);

    return 0;
}

```

P502

DIFFICUTLY ★★★

编写一程序实现以下功能:

1. 把 ASC 字符的点阵字库 ASC12.fon (见附件) 保存到程序 P502.c 所在的文件夹且文件名保持不变。ASC12.fon 文件格式说明: 字符的点阵数据存储在文件 ASC12.fon 时, 按字符的 ASCII 码 (0 ~ 255) 为序由前到后存储; 字符为 12 (高) × 8 (宽) 点阵, 存储时每行 8 点为 1 字节 (相应位为 1 表示有点、为 0 表示无点), 从第 1 行到第 12 行按序存放。
2. 程序运行时先显示 “Please input a char:”, 等待用户从键盘输入任一字符 (例如, 如图 502.1 中所示输入 “A”) 并按回车后, 从前面给出的点阵字库文件中取出该字符的点阵数据。
3. 将该字符的点阵数据在屏幕上加框显示: 相应位置有点则显示 “\”, 无点则显示 “ ” (空格), 并且字符外围加框。

可用素材: `printf("Please input a char:");`

程序的运行效果应严格地如图 502.1 所示, 金色部分是从键盘输入的内容。



图 502.1 “A” 的字模显示

附件:  ASC12.fon

参考答案:

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

```
void Read(char font[], int ch, FILE *fp);
```

```
void Display(char font[]);
```

```
int main(void)
```

```
{
```

```
    FILE *fp;
```

```
    char ch, font[16];
```

```
    fp = fopen("ASC12.fon", "rb");
```

```
    if (fp == NULL)
```

```
    {
```

```
        printf("File open error!\n");
```

```
        return 0;
```

```
    }
```

```
    printf("Please input a char:");
```

```
    ch = getchar();
```

```
    putchar('\n');
```

```
    Read(font, ch, fp);
```

```
    Display(font);
```

```
    fclose(fp);
```

```
    return 0;
```

```
}
```

```

void Read(char font[], int ch, FILE *fp)
{
    int input;

    input = ch;
    input *= 12;
    fseek(fp, input, SEEK_SET);
    fread(font, 1, 12, fp);
}

void Display(char font[])
{
    int i, k, bit;

    for (i = 0; i < 12; i++)
    {
        if (i == 0 || i == 11)
        {
            font[i] |= 0xff;
        }
        font[i] |= 0x81;
        bit = 1 << 7;
        for (k = 0; k < 8; k++)
        {
            if (font[i] & bit)
            {
                putchar('\\');
            }
            else
            {
                putchar(' ');
            }
            bit >>= 1;
        }
        putchar('\n');
    }
}

```

P766

DIFFICUTLY ★★★

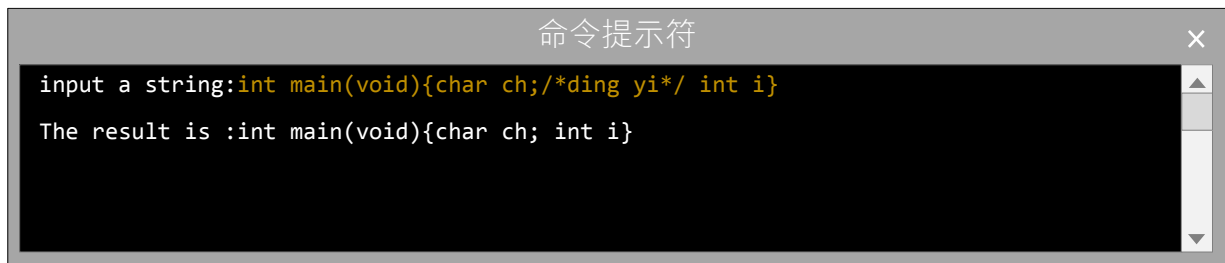
从键盘上读入一行字符（约定：字符数 ≤ 127 字节），判断其中的注释是否合法，不合法则报错，合法时则删除注释后再输出。合法注释是指“/*”标记注释开始、“*/”标记注释结束，通常表现为“/*...*/”。

注意事项：

1. 程序中不能使用库函数 strstr() 或使用同名的变量、函数、单词。
2. 只考虑行内最多只包含一个注释的情况。
3. 不合法的注释情况有很多种，例如：“...*/”缺注释开始标记、“/*...*/”注释开始标记错误、“/*...”缺注释结束标记、“/*...*/”注释结束标记错误。

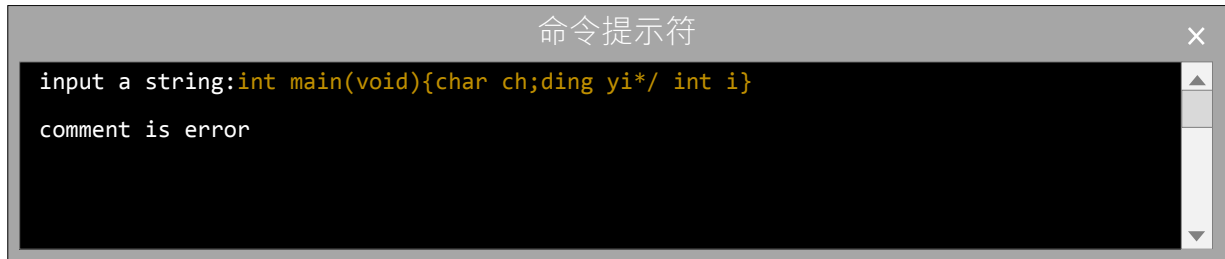
可用素材： printf("input a string:")
printf("\nThe result is :")
printf("\ncomment is error\n")

程序的运行效果应类似地如图 766.1、图 766.2、图 766.3 和图 766.4 所示，金色部分是从键盘输入的内容。



```
命令提示符
input a string:int main(void){char ch; /*ding yi*/ int i}
The result is :int main(void){char ch; int i}
```

图 766.1



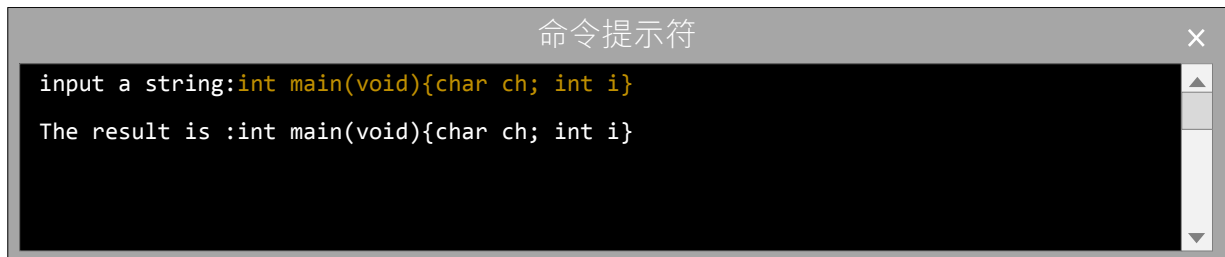
```
命令提示符
input a string:int main(void){char ch; ding yi*/ int i}
comment is error
```

图 766.2



```
命令提示符
input a string:int main(void){char ch; /*ding yi* / int i}
comment is error
```

图 766.3



```
命令提示符
input a string:int main(void){char ch; int i}
The result is :int main(void){char ch; int i}
```

图 766.4

参考答案:

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

```
int main(void)
{
    char str[128], result[128];
    int i, j, point1, point2, flag1 = 0, flag2 = 0;

    printf("input a string:");
    gets(str);

    for (i = 0; str[i] != '\0'; i++)
    {
        if (str[i] == '/' && str[i + 1] == '*')
        {
            point1 = i;
            flag1 = 1;
            break;
        }
    }
}
```

```

for (i = 0; str[i] != '\0'; i++)
{
    if (str[i] == '*' && str[i + 1] == '/')
    {
        point2 = i;
        flag2 = 1;
        break;
    }
}

if (flag1 == 0 && flag2 == 0)
{
    printf("\nThe result is :");
    puts(str);
}
else if (flag1 == 1 && flag2 == 1 && point1 < point2)
{
    printf("\nThe result is :");
    for (i = 0, j = 0; str[i] != '\0'; i++)
    {
        if (i < point1 || i > point2 + 1)
        {
            result[j] = str[i];
            j++;
        }
    }
    result[j] = '\0';
    puts(result);
}
else
{
    printf("\ncomment is error\n");
}

return 0;
}

```

P781

DIFFICUTLY ★★★

从键盘上输入 5 个字符串（约定：每个字符串中字符数 ≤ 80 字节），对其进行升序排序并输出。

注：程序中不能使用库函数 strcpy()、strcat()、strncat()、strncpy()、memcpy()、strcpy()、memcpy()、memcmp()、strcmp()、strncmp()、strncmpi()、strnicmp()、strcmp() 和 strcmpi() 或使用同名的变量、函数、单词。

可用素材： printf("Input 5 strings:\n")
printf("-----\n")

程序的运行效果应类似地如图 781.1 所示，金色部分是从键盘输入的内容。



图 781.1

参考答案:

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

```
void MyStrcpy(char str1[], char str2[]);
```

```
int MyStrcmp(char str1[], char str2[]);
```

```
int main(void)
```

```
{
```

```
    char str[5][81], tmp[81];
```

```
    int i, j;
```

```
    printf("Input 5 strings:\n");
```

```
    for (i = 0; i < 5; i++)
```

```
    {
```

```
        gets(str[i]);
```

```
    }
```

```
    for (j = 1; j < 5; j++)
```

```
    {
```

```
        for (i = 0; i < 5 - j; i++)
```

```
        {
```

```
            if (MyStrcmp(str[i], str[i + 1]) > 0)
```

```
            {
```

```
                MyStrcpy(tmp, str[i]);
```

```
                MyStrcpy(str[i], str[i + 1]);
```

```
                MyStrcpy(str[i + 1], tmp);
```

```
            }
```

```
        }
```

```
    }
```

```
    printf("-----\n");
```

```
    for (i = 0; i < 5; i++)
```

```
    {
```

```
        printf("%s\n", str[i]);
```

```
    }
```

```
    return 0;
```

```
}
```

```
int MyStrcmp(char str1[], char str2[])
```

```
{
```

```
    int i, flag = 0;
```

```
    for (i = 0; ; i++)
```

```
    {
```

```
        if (str1[i] > *(str2 + i) || *(str1 + i) != '\0' && *(str2 + i) == '\0')
```

```
        {
```

```
            flag = 1;
```

```

        break;
    }
    else if (*(str1 + i) < *(str2 + i) || *(str1 + i) == '\0' && *(str2 + i) != '\0')
    {
        flag = -1;
        break;
    }
    else if (*(str1 + i) == '\0' && *(str2 + i) == '\0')
    {
        break;
    }
}

return flag;
}

void MyStrcpy(char str1[], char str2[])
{
    while (*str2 != '\0')
    {
        *str1 = *str2;
        str1++;
        str2++;
    }
    *str1 = '\0';
}

```

P794

DIFFICUTLY ★★★

程序 P794.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 换成代码）。

程序的功能是：调用自定义函数 creat() 读入第一部分学生信息建立链表 A、调用自定义函数 creat() 读入第二部分学生信息建立链表 B，每个链表中的节点包括学号、成绩；然后调用自定义函数 merge() 将链表 B 链接到链表 A 后。三个链表的内容均调用自定义函数 print() 输出在屏幕上。

可用素材： printf("学生 %d: "...

程序的运行效果应类似地如图 794.1 所示，金色部分是从键盘输入的内容。

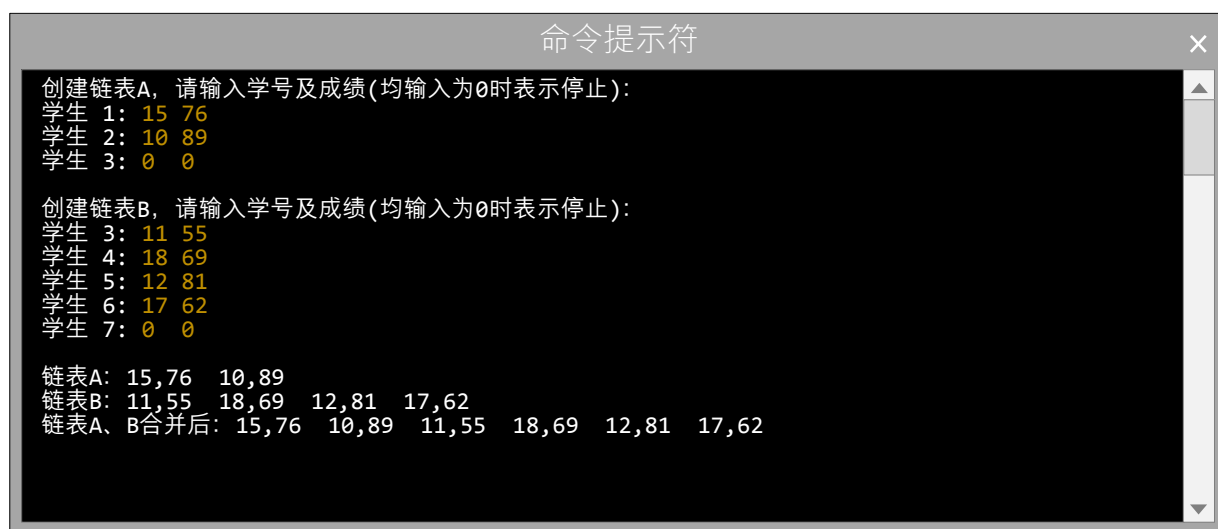


图 794.1

附件:  P794.c

文件内容:

P794.c

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

/* User Code Begin(考生可在本行后添加代码, 定义程序中使用的结构体类型、声明自定义函数的原型, 行数不限) */
<A>
/* User Code End(考生添加代码结束) */

/* print以规定的格式完成遍历显示指定的链表 */
void print(char *Info, struct student *Head);

int main(void)
{
    struct student *ah, *bh;

    printf("创建链表A, 请输入学号及成绩(均输入为0时表示停止): \n");
    ah = creat();
    printf("\n创建链表B, 请输入学号及成绩(均输入为0时表示停止): \n");
    bh = creat();

    print("\n链表A: ", ah);
    print("\n链表B: ", bh);

    ah = merge(ah, bh);
    print("\n链表A、B合并后: ", ah);

    return 0;
}

void print(char *Info, struct student *Head)
{
    printf("%s", Info);
    while (Head != NULL)
    {
        printf("%d,%d ", Head->num, Head->score);
        Head = Head->next;
    }
}

/* User Code Begin(考生在此后完成自定义函数的设计, 行数不限) */
<B>
```

参考答案:

```
<A> struct student
{
    int num, score;
    struct student *next;
};

struct student *creat(void);
struct student *merge(struct student *ah, struct student *bh);
<B> struct student *creat(void)
{
    struct student *Head, *p1, *p2;
```

```

static int count;
int flag = 0;

Head = p1 = p2 = (struct student*)malloc(sizeof(struct student));
while (1)
{
    count++;
    printf("学生 %d: ", count);
    scanf("%d%d", &p1->num, &p1->score);
    if (p1->num == 0 && p1->score == 0)
    {
        break;
    }

    if (Head == NULL)
    {
        Head = p1;
    }
    else
    {
        p2->next = p1;
    }
    p2 = p1;
    p1 = (struct student *)malloc(sizeof(struct student));
    flag = 1;
}
count--;
p2->next = NULL;

if (flag == 0)
{
    return NULL;
}
return Head;
}

struct student *merge(struct student *ah, struct student *bh)
{
    struct student *Head, *p0;

    if (ah == NULL && bh == NULL)
    {
        return NULL;
    }
    else if (ah != NULL && bh == NULL)
    {
        return ah;
    }
    else if (ah == NULL && bh != NULL)
    {
        return bh;
    }

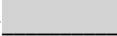
    Head = p0 = ah;
    while (p0->next != NULL)
    {
        p0 = p0->next;
    }
    p0->next = bh;

    return Head;
}

```

P316


DIFFICUTLY ★★★★★

程序 P316.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将  换成代码）。

程序的运行效果应类似地如图 316.1 所示，金色部分为从键盘输入的内容。注意，在“input string-2:”处输入时直接按了回车键，这是题目所允许的。



图 316.1

附件:  P316.c

文件内容:

P316.c

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

/* GetString的功能是从键盘读入最多n个字符(遇到由endFlag指定的字符或回车则结束读并且该字符和回车
   均视为有效字符)存放在buf所指向的内存中并加上字符串结束符'\0'
   函数返回值为读入的字符个数 */
int GetString(char *buf, char endFlag, int n);

/* CompDigit 的功能是分别统计指针s1、s2、s3、s4所指向的字符串中数字字符的个数,
   将包含最多数字字符(不考虑存在包含相同个数数字字符的情况)的字符串的长度放在指针mLen所指向的
   内存单元中,并通过函数返回值返回该字符串中第1个数字字符的地址 */
char *CompDigit(char *s1, char *s2, char *s3, char *s4, int *mLen);

/* clearRestInput的功能是将前一次读字符后到回车(含回车)之间的字符废弃 */
void clearRestInput(char lastInput);

/* 本部分代码功能建议: 考生新增函数原型声明(非必须新增) */
/* User Code Begin(Limit: lines<=2, lineLen<=80, 考生可在本行后添加代码、最多2行、每行长<=80字符) */
<A>
/* User Code End(考生添加代码结束。注意: 空行和单独为一行的{与}均不计行数、行长不计行首tab缩进) */

int main(void)
{
    char str[4][30], strM[30], endChar, lastChar;
    int totalLen = 0, maxLen = -1, i;
    const int N = 29;

    printf("\ninput endFlag char: ");
    scanf("%c", &endChar);

    for (i=0, lastChar=endChar; i<4; i++)
    {
        clearRestInput(lastChar);
```

```

        printf("input string-%d: ", i+1);
        totalLen += GetString(str[i], endChar, N);
        lastChar = str[i][strlen(str[i]) - 1];
    }

    strcpy(strM, CompDigit(str[0], str[1], str[2], str[3], &maxLen));
    printf("\n-----\n");
    printf("%s, 4Len=%d, mLen=%d\n", strM, totalLen, maxLen);

    return 0;
}

void clearRestInput(char lastInput)
{
    while (lastInput != '\n')
    {
        lastInput = getchar();
    }
}

/* User Code Begin(考生在此后完成函数GetString和CompDigit的设计, 在其中不要使用printf函数, 行数不限) */
<B>

```

参考答案:

```

<A> /* Blank */
<B> int GetString(char *buf, char endFlag, int n)
{
    int num = 0;
    char ch;

    scanf("%c", &ch);
    while (ch != endFlag && ch != '\n')
    {
        *buf = ch;
        buf++;
        num++;
        if (num >= n)
        {
            break;
        }
        scanf("%c", &ch);
    }
    if (ch == endFlag || ch == '\n')
    {
        *buf = ch;
        buf++;
        num++;
    }
    *buf = '\0';

    return num;
}

char *CompDigit(char *s1, char *s2, char *s3, char *s4, int *mLen)
{
    int i, num[4] = { 0 }, digitnum;
    char *str[4], *maxStr;

    for (i = 0; s1[i] != '\0'; i++)
    {

```



```

        if (s1[i] >= '0' && s1[i] <= '9')
        {
            num[0]++;
        }
    }
    for (i = 0; s2[i] != '\0'; i++)
    {
        if (s2[i] >= '0' && s2[i] <= '9')
        {
            num[1]++;
        }
    }
    for (i = 0; s3[i] != '\0'; i++)
    {
        if (s3[i] >= '0' && s3[i] <= '9')
        {
            num[2]++;
        }
    }
    for (i = 0; s4[i] != '\0'; i++)
    {
        if (s4[i] >= '0' && s4[i] <= '9')
        {
            num[3]++;
        }
    }

    str[0] = s1;
    str[1] = s2;
    str[2] = s3;
    str[3] = s4;
    digitnum = num[0];
    maxStr = str[0];
    for (i = 1; i < 4; i++)
    {
        if (num[i] > digitnum)
        {
            digitnum = num[i];
            maxStr = str[i];
            *mLen = strlen(str[i]);
        }
    }
    while (1)
    {
        if (*maxStr >= '0' && *maxStr <= '9')
        {
            return maxStr;
        }
        maxStr++;
    }
}

```

P507

DIFFICUTLY ★★★★★ | NO REMARKS

系统已给出示例参考程序 P507.c（见文件内容），要求编写出的程序的运行效果与程序 P507.exe（见附件）基本相同。

附件:  

P507.c P507.exe

文件内容:

P507.c

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

HANDLE hCon;

void SetColor(int ForeColor, int BackGroundColor); //设置显示字符的前景和背景颜色, 取值范围0-15
void gotoRC(int row, int col); //设置光标位置, row为行座标取值0-24, col为列座标取值0-79

int main(void)
{
    int i;

    hCon = GetStdHandle(STD_OUTPUT_HANDLE);

    SetColor(0, 8+1);
    for (i=0; i<10; i++)
    {
        gotoRC(10, 30+i*2);
        putchar(' ');
        Sleep(200); //休眠或暂停本程序执行200毫秒
    }

    SetColor(14, 0);
    for (i=0; i<10; i++)
    {
        gotoRC(13, 30+i*2);
        putchar('$');
        Sleep(200);
    }

    gotoRC(20, 0);
    printf("Example End!\n");
    return 0;
}

void SetColor(int ForeColor, int BackGroundColor)
{
    SetConsoleTextAttribute(hCon, (unsigned short)(ForeColor|(BackGroundColor*16)));
}

void gotoRC(int row, int col)
{
    COORD pos = {col, row};

    SetConsoleCursorPosition(hCon, pos);
}
```

参考答案:

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

HANDLE hCon;

void SetColor(int ForeColor, int BackGroundColor);
void gotoRC(int row, int col);

int main(void)
{
```

```

int i, j, k;

hCon = GetStdHandle(STD_OUTPUT_HANDLE);

SetColor(0, 9);
for (i = 0; i < 25; i++)
{
    for (j = 0; j < 80; j++)
    {
        gotoRC(i, j);
        putchar(' ');
    }
    Sleep(20); /* 设置休眠时间, 可自定义, 下同 */
}

SetColor(0, 12);
for (i = 0; i < 80; i++)
{
    for (j = 0; j < 25; j++)
    {
        gotoRC(j, i);
        putchar(' ');
    }
    Sleep(20);
}

SetColor(0, 14);
for (i = 79; i >= 0; i--)
{
    for (j = 0; j < 25; j++)
    {
        gotoRC(j, i);
        putchar(' ');
    }
    Sleep(20);
}

SetColor(0, 10);
for (i = 11, k = 12; i >= 0 || k == 24; i--, k++)
{
    for (j = 0; j < 80; j++)
    {
        gotoRC(i, j);
        putchar(' ');
        gotoRC(k, j);
        putchar(' ');
    }
    Sleep(40);
}

SetColor(0, 0);
for (i = 39, k = 40; i >= 0; i--, k++)
{
    for (j = 0; j < 25; j++)
    {
        gotoRC(j, i);
        putchar(' ');
        gotoRC(j, k);
        putchar(' ');
    }
    Sleep(40);
}

```

```

SetColor(14, 0);
for (i = 0; i < 80; i++)
{
    if (i == 78)
    {
        gotoRC(0, 78);
        printf("***");
        gotoRC(1, 79);
        printf("o");
        Sleep(100);
        system("cls");
    }
    else if (i == 79)
    {
        gotoRC(0, 79);
        printf("*");
        gotoRC(1, 79);
        printf("*");
        gotoRC(2, 79);
        printf("o");
        Sleep(100);
        system("cls");
    }
    else
    {
        gotoRC(0, i);
        printf("***o");
        Sleep(100);
        system("cls");
    }
}

for (j = 3; j < 24; j++)
{
    if (j == 22)
    {
        gotoRC(22, 79);
        printf("*");
        gotoRC(23, 79);
        printf("*");
        gotoRC(23, 78);
        printf("o");
        Sleep(100);
        system("cls");
    }
    else if (j == 23)
    {
        gotoRC(22, 79);
        printf("*");
        gotoRC(23, 78);
        printf("o*");
        Sleep(100);
        system("cls");
    }
    else
    {
        gotoRC(j, 79);
        printf("*");
        gotoRC(j + 1, 79);
        printf("*");
        gotoRC(j + 2, 79);
        printf("o");
        Sleep(100);
    }
}

```

```

        system("cls");
    }
}

for (i = 76; i >= 0; i--)
{
    if (i == 1)
    {
        gotoRC(22, 0);
        printf("***");
        gotoRC(21, 0);
        printf("o");
        Sleep(100);
        system("cls");
    }
    else if (i == 0)
    {
        gotoRC(22, 0);
        printf("*");
        gotoRC(21, 0);
        printf("*");
        gotoRC(20, 0);
        printf("o");
        Sleep(100);
        system("cls");
    }
    else
    {
        gotoRC(22, i);
        printf("o*");
        Sleep(100);
        system("cls");
    }
}

for (j = 19; j >= 0; j--)
{
    if (j == 1)
    {
        gotoRC(0, 0);
        printf("*o");
        gotoRC(1, 0);
        printf("*");
        Sleep(100);
        system("cls");
    }
    else if (j == 0)
    {
        gotoRC(0, 0);
        printf("***o");
    }
    else
    {
        gotoRC(j, 0);
        printf("o");
        gotoRC(j + 1, 0);
        printf("*");
        gotoRC(j + 2, 0);
        printf("*");
        Sleep(100);
        system("cls");
    }
}

```

```

    gotoRC(12, 40); /* 设置光标位置, 可自定义 */
    Sleep(6000);

    return 0;
}

void SetColor(int ForeColor, int BackGroundColor)
{
    SetConsoleTextAttribute(hCon, (unsigned short)(ForeColor | (BackGroundColor * 16)));
}

void gotoRC(int row, int col)
{
    COORD pos = { col, row };

    SetConsoleCursorPosition(hCon, pos);
}

```

P511

DIFFICUTLY ★★★★★ | C/C++ | NO REMARKS

通过猪的安家计算有多少头猪。Andy 和 Mary 养了很多猪，他们想要给猪安家。但是 Andy 没有足够的猪圈，很多猪只能在一个猪圈安家。举个例子，假如有 16 头猪，Andy 建了 3 个猪圈，为了保证公平，剩下 1 头猪就没有地方安家了。Mary 生气了，骂 Andy 没有脑子，并让他重新建立猪圈。这回 Andy 建造了 5 个猪圈，但是仍然有 1 头猪没有地方去，然后 Andy 又建造了 7 个猪圈，但是还有 2 头没有地方去。Andy 都快疯了。你对这个事情感兴趣起来，你想通过 Andy 建造猪圈的过程，知道 Andy 家至少养了多少头猪。

程序输入：包含多组建造猪圈测试数据。每组数据第一行包含一个整数 n （约定 $n \leq 10$ ），表示 Andy 建立猪圈的次数，接下来 n 行，每行两个整数 a_i, b_i （约定 $b_i \leq a_i \leq 1000$ ），表示 Andy 建立了 a_i 个猪圈时有 b_i 头猪没有去处。

程序输出：输出包含一个正整数，即为 Andy 家至少养猪的数目。

程序的运行效果应类似地如图 511.1 所示，金色部分是从键盘输入内容。

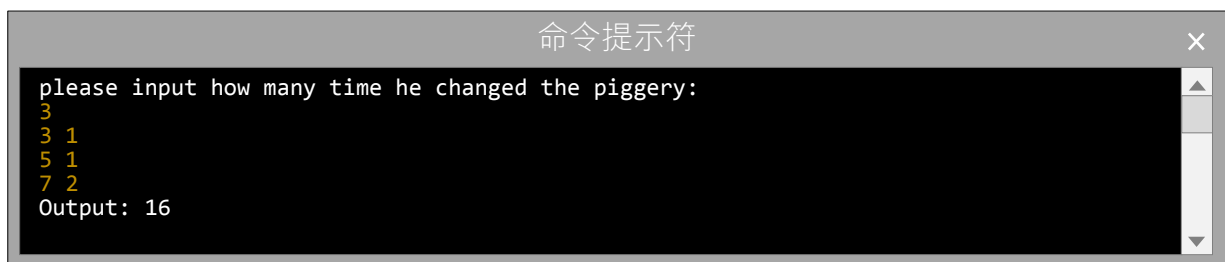


图 511.1

参考答案:

```

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

struct node
{
    int ai, bi;
    struct node *next;
};

struct node *creatfun(int n);
void workfun(struct node *head, int n);

```

```

int main(void)
{
    struct node *p0;
    int n;

    printf("please input how many time he changed the piggery:\n");
    scanf("%d", &n);
    p0 = creatfun(n);
    workfun(p0, n);
    return 0;
}

struct node *creatfun(int n)
{
    struct node *head, *p0, *s;
    int count;

    p0 = (struct node *)malloc(sizeof(struct node));
    s = head = p0;

    scanf("%d%d", &p0->ai, &p0->bi);
    for (count = 1; count < n; count++)
    {
        p0 = (struct node *)malloc(sizeof(struct node));
        s->next = p0;
        scanf("%d%d", &p0->ai, &p0->bi);
        s = p0;
    }
    p0->next = NULL;

    return head;
}

void workfun(struct node *head, int n)
{
    struct node *p0;
    int count, max, right = 0;

    p0 = head;
    max = p0->ai + p0->bi;

    while (p0 != NULL)
    {
        if ((p0->ai + p0->bi) > max)
        {
            max = p0->ai + p0->bi;
        }
        p0 = p0->next;
    }
    p0 = head;

    for (count = max; ; count++)
    {
        while (p0 != NULL)
        {
            if ((count % (p0->ai) == p0->bi) && (count / (p0->ai)) > 0)
            {
                right++;
            }
            p0 = p0->next;
        }
        if (right == n)
        {

```

```

        printf("Output: %d\n", count);
        break;
    }
    else
    {
        right = 0;
        p0 = head;
    }
}
}

```

P512

DIFFICUTLY ★★★★★ | C/C++ | NO REMARKS

保持数列有序。有 n (约定 $n \leq 100$) 个整数，已经按照从小到大顺序排列好，现在另外给一个整数 x ，请将该数插入到序列中，并使新的序列仍然有序。

程序输入：输入数据包含多组测试实例，每组数据由两行组成，第一行是 n 和 x ，第二行是已经有序的 n 个数的数列。 n 和 x 同时为 0 表示输入数据的结束，本行不做处理。

程序输出：对于每个测试实例，输出插入新的元素后的数列。

程序的运行效果应类似地如图 512.1 所示，金色部分是从键盘输入内容。



图 512.1

参考答案：

```

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

int cmp(const void *p1, const void *p2);

int main(void)
{
    int n0, m0, i, arr[100];

    printf("Input:\n");
    while (scanf("%d%d", &n0, &m0) != EOF)
    {
        if (n0 == 0 && m0 == 0)
        {
            break;
        }
        arr[n0] = 0;

        for (i = 0; i < n0; i++)
        {
            scanf("%d", &arr[i]);
        }
    }
}

```



```

arr[n0] = m0;
qsort(arr, n0 + 1, sizeof(arr[0]), cmp);

printf("Output: ");
for (i = 0; i < n0 + 1; i++)
{
    if (i == 0)
    {
        printf("%d", arr[0]);
    }
    else
    {
        printf(" %d", arr[i]);
    }
}
putchar('\n');
}

return 0;
}

int cmp(const void *p1, const void *p2)
{
    return *(int *)p1 - *(int *)p2;
}

```

P513

DIFFICUTLY ★★★★★ | C/C++ | NO REMARKS

在平面上的 n 个点中找出横坐标最小的点。

程序输入：第一行是一个正整数 n （约定 $1 \leq n \leq 1000$ ），表示一共有 n 个点。接下来的 n 行，每行 2 个整数，分别表示每个点的横坐标和纵坐标。

程序输出：输出横坐标最小的那个点的横坐标和纵坐标，如果横坐标最小的点有多个，则只输出其中纵坐标最小的那个点。

程序的运行效果应类似地如图 513.1 所示，金色部分是从键盘输入内容。



图 513.1

参考答案 (C++) :

```

P513.cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

struct node
{

```

```

    int x;
    int y;
};

bool iscmpx(node *a, node *b);
bool iscmpy(node *a, node *b);

int main(void)
{
    int n;

    cout << "Input:" << endl;
    cin >> n;
    vector < node * > nvec;
    for (int i = 0; i < n; i++)
    {
        node *nd = new node;
        cin >> nd->x >> nd->y;
        nvec.push_back(nd);
    }

    sort(nvec.begin(), nvec.end(), iscmpy);
    stable_sort(nvec.begin(), nvec.end(), iscmpx);
    cout << "Output: " << nvec.front()->x << " " << nvec.front()->y << endl;

    return 0;
}

bool iscmpx(node *a, node *b)
{
    return a->x < b->x;
}

bool iscmpy(node *a, node *b)
{
    return a->y < b->y;
}

```

P514

DIFFICUTLY ★★★★★ | C/C++ | NO REMARKS

赫夫曼编码。给出 n 个有权值的结点，构造赫夫曼树，输出所有这 n 个结点的权值与其赫曼编码长度（即该结点在所构造的赫曼树中的深度）的乘积的总和。

程序输入：第一行是一个正整数 n ，表示一共有 n 个结点。第二行为 n 个整数，分别表示这 n 个结点的权值。

程序输出：输出所有这 n 个结点的权值与其赫曼编码长度的乘积的总和。

程序的运行效果应类似地如图 514.1 所示，金色部分是从键盘输入内容。



图 514.1

参考答案 (C++) :

P514.cpp

```
#include <iostream>
```

```
#include <list>
```

```
#include <algorithm>
```

```
using namespace std;
```

```
int len = 0;
```

```
struct node
```

```
{
```

```
    int value;
```

```
    int depth;
```

```
    node *left;
```

```
    node *right;
```

```
    bool operator < (node &a0)
```

```
    {
```

```
        return value < a0.value;
```

```
    }
```

```
};
```

```
void calculate(node *n);
```

```
void getSum(node *n, int &s);
```

```
int main(void)
```

```
{
```

```
    int n;
```

```
    int sum = 0;
```

```
    cout << "Input:" << endl;
```

```
    cin >> n;
```

```
    list < node * > nlist;
```

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

```
    {
```

```
        node *nd = new node;
```

```
        cin >> nd->value;
```

```
        nd->left = nd->right = 0;
```

```
        nlist.push_back(nd);
```

```
    }
```

```
    nlist.sort();
```

```
    node *root;
```

```
    while (nlist.size() > 1)
```

```
    {
```

```
        node *nd = new node;
```

```
        node *nd1 = nlist.front();
```

```
        nlist.pop_front();
```

```
        node *nd2 = nlist.front();
```

```
        nlist.pop_front();
```

```
        nd->value = nd1->value + nd2->value;
```

```
        nd->left = nd1;
```

```
        nd->right = nd2;
```

```
        list < node * > ::iterator it;
```

```
        for (it = nlist.begin(); it != nlist.end(); it++)
```

```
        {
```

```
            if ((*it)->value > nd->value)
```

```
            {
```

```
                break;
```

```
            }
```

```

        }
        nlist.insert(it, nd);
    }
    root = nlist.front();

    calculate(root);
    getSum(root, sum);
    cout << "Output: " << sum << endl;
    return 0;
}

void calculate(node *n)
{
    if (n != 0)
    {
        n->depth = len;
        len++;
        calculate(n->left);
        calculate(n->right);
        len--;
    }
}

void getSum(node *n, int &s)
{
    if (n != 0)
    {
        if (n->left == 0 && n->right == 0)
        {
            s += (n->depth) * (n->value);
        }
        if (n->left != 0)
        {
            getSum(n->left, s);
        }
        if (n->right != 0)
        {
            getSum(n->right, s);
        }
    }
}

```

P515

DIFFICUTLY ★★★★★ | C/C++ | NO REMARKS

子矩阵旋转操作。

程序输入：前 5 行每行 5 个整数，表示一个 5×5 矩阵。第 6 行为 4 个整数 a, b, c, d ，根据这 4 个整数进行相应的旋转操作。

程序输出：

1. 若 a, b 分别为 1, 2，则将原矩阵中以第 c 行第 d 列元素为左上角的 2×2 子矩阵顺时针旋转 90 度后输出原矩阵。
2. 若 a, b 分别为 1, 3，则将原矩阵中以第 c 行第 d 列元素为左上角的 3×3 子矩阵顺时针旋转 90 度后输出原矩阵。
3. 若 a, b 分别为 2, 2，则将原矩阵中以第 c 行第 d 列元素为左上角的 2×2 子矩阵逆时针旋转 90 度后输出原矩阵。

4. 若 a, b 分别为 2, 3, 则将原矩阵中以第 c 行第 d 列元素为左上角的 3×3 子矩阵逆时针旋转 90 度后输出原矩阵。

程序的运行效果应类似地如图 515.1 所示, 金色部分是从键盘输入内容。



图 515.1

参考答案 1 (c) :

P515.c

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

```
int main(void)
```

```
{
```

```
    int alt[5][5], arr1[4], arr2[9], i, j, cmdx, cmdy, x0, y0;
```

```
    printf("Input:\n");
```

```
    for (i = 0; i < 5; i++)
```

```
    {
```

```
        for (j = 0; j < 5; j++)
```

```
        {
```

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

```
        }
```

```
    }
```

```
    scanf("%d%d%d%d", &cmdx, &cmdy, &x0, &y0);
```

```
    if (cmdx == 2)
```

```
    {
```

```
        if (cmdy == 2)
```

```
        {
```

```
            arr1[0] = alt[x0 - 1][y0 - 1];
```

```
            arr1[1] = alt[x0 - 1][y0];
```

```
            arr1[2] = alt[x0][y0 - 1];
```

```
            arr1[3] = alt[x0][y0];
```

```
            alt[x0 - 1][y0 - 1] = arr1[1];
```

```
            alt[x0 - 1][y0] = arr1[3];
```

```
            alt[x0][y0 - 1] = arr1[0];
```

```
            alt[x0][y0] = arr1[2];
```

```
        }
```

```
        else
```

```
        {
```

```
            arr2[0] = alt[x0 - 1][y0 - 1];
```

```
            arr2[1] = alt[x0 - 1][y0];
```

```
            arr2[2] = alt[x0 - 1][y0 + 1];
```

```
            arr2[3] = alt[x0][y0 - 1];
```

```
            arr2[4] = alt[x0][y0];
```

```
            arr2[5] = alt[x0][y0 + 1];
```

```
            arr2[6] = alt[x0 + 1][y0 - 1];
```

```
            arr2[7] = alt[x0 + 1][y0];
```

```
            arr2[8] = alt[x0 + 1][y0 + 1];
```

```

        alt[x0 - 1][y0 - 1] = arr2[2];
        alt[x0 - 1][y0] = arr2[5];
        alt[x0 - 1][y0 + 1] = arr2[8];
        alt[x0][y0 - 1] = arr2[1];
        alt[x0][y0] = arr2[4];
        alt[x0][y0 + 1] = arr2[7];
        alt[x0 + 1][y0 - 1] = arr2[0];
        alt[x0 + 1][y0] = arr2[3];
        alt[x0 + 1][y0 + 1] = arr2[6];
    }
}
else
{
    if (cmdy == 2)
    {
        arr1[0] = alt[x0 - 1][y0 - 1];
        arr1[1] = alt[x0 - 1][y0];
        arr1[2] = alt[x0][y0 - 1];
        arr1[3] = alt[x0][y0];
        alt[x0 - 1][y0 - 1] = arr1[2];
        alt[x0 - 1][y0] = arr1[0];
        alt[x0][y0 - 1] = arr1[3];
        alt[x0][y0] = arr1[1];
    }
    else
    {
        arr2[0] = alt[x0 - 1][y0 - 1];
        arr2[1] = alt[x0 - 1][y0];
        arr2[2] = alt[x0 - 1][y0 + 1];
        arr2[3] = alt[x0][y0 - 1];
        arr2[4] = alt[x0][y0];
        arr2[5] = alt[x0][y0 + 1];
        arr2[6] = alt[x0 + 1][y0 - 1];
        arr2[7] = alt[x0 + 1][y0];
        arr2[8] = alt[x0 + 1][y0 + 1];
        alt[x0 - 1][y0 - 1] = arr2[6];
        alt[x0 - 1][y0] = arr2[3];
        alt[x0 - 1][y0 + 1] = arr2[0];
        alt[x0][y0 - 1] = arr2[7];
        alt[x0][y0] = arr2[4];
        alt[x0][y0 + 1] = arr2[1];
        alt[x0 + 1][y0 - 1] = arr2[8];
        alt[x0 + 1][y0] = arr2[5];
        alt[x0 + 1][y0 + 1] = arr2[2];
    }
}

printf("Output:\n");
for (i = 0; i < 5; i++)
{
    for (j = 0; j < 4; j++)
    {
        printf("%-3d", alt[i][j]);
    }
    printf("%d\n", alt[i][4]);
}

return 0;
}

```

参考答案 2 (C++) :

P515.cpp

```
#include <iostream>
```

```
using namespace std;
```

```
int main(void)
```

```
{
```

```
    int alt[5][5];
```

```
    int cmdx, cmdy, x, y;
```

```
    cout << "Input:" << endl;
```

```
    for (int i = 0; i < 5; i++)
```

```
    {
```

```
        for (int j = 0; j < 5; j++)
```

```
        {
```

```
            cin >> alt[i][j];
```

```
        }
```

```
    }
```

```
    cin >> cmdx >> cmdy >> x >> y;
```

```
    if (cmdx == 2)
```

```
    {
```

```
        if (cmdy == 2)
```

```
        {
```

```
            int *arr = new int[4];
```

```
            arr[0] = alt[x - 1][y - 1];
```

```
            arr[1] = alt[x - 1][y];
```

```
            arr[2] = alt[x][y - 1];
```

```
            arr[3] = alt[x][y];
```

```
            alt[x - 1][y - 1] = arr[1];
```

```
            alt[x - 1][y] = arr[3];
```

```
            alt[x][y - 1] = arr[0];
```

```
            alt[x][y] = arr[2];
```

```
        }
```

```
    else
```

```
    {
```

```
        int *arr = new int[9];
```

```
        arr[0] = alt[x - 1][y - 1];
```

```
        arr[1] = alt[x - 1][y];
```

```
        arr[2] = alt[x - 1][y + 1];
```

```
        arr[3] = alt[x][y - 1];
```

```
        arr[4] = alt[x][y];
```

```
        arr[5] = alt[x][y + 1];
```

```
        arr[6] = alt[x + 1][y - 1];
```

```
        arr[7] = alt[x + 1][y];
```

```
        arr[8] = alt[x + 1][y + 1];
```

```
        alt[x - 1][y - 1] = arr[2];
```

```
        alt[x - 1][y] = arr[5];
```

```
        alt[x - 1][y + 1] = arr[8];
```

```
        alt[x][y - 1] = arr[1];
```

```
        alt[x][y] = arr[4];
```

```
        alt[x][y + 1] = arr[7];
```

```
        alt[x + 1][y - 1] = arr[0];
```

```
        alt[x + 1][y] = arr[3];
```

```
        alt[x + 1][y + 1] = arr[6];
```

```
    }
```

```
}
```

```
else
```

```
{
```

```

    if (cmdy == 2)
    {
        int *arr = new int[4];

        arr[0] = alt[x - 1][y - 1];
        arr[1] = alt[x - 1][y];
        arr[2] = alt[x][y - 1];
        arr[3] = alt[x][y];
        alt[x - 1][y - 1] = arr[2];
        alt[x - 1][y] = arr[0];
        alt[x][y - 1] = arr[3];
        alt[x][y] = arr[1];
    }
    else
    {
        int *arr = new int[9];

        arr[0] = alt[x - 1][y - 1];
        arr[1] = alt[x - 1][y];
        arr[2] = alt[x - 1][y + 1];
        arr[3] = alt[x][y - 1];
        arr[4] = alt[x][y];
        arr[5] = alt[x][y + 1];
        arr[6] = alt[x + 1][y - 1];
        arr[7] = alt[x + 1][y];
        arr[8] = alt[x + 1][y + 1];
        alt[x - 1][y - 1] = arr[6];
        alt[x - 1][y] = arr[3];
        alt[x - 1][y + 1] = arr[0];
        alt[x][y - 1] = arr[7];
        alt[x][y] = arr[4];
        alt[x][y + 1] = arr[1];
        alt[x + 1][y - 1] = arr[8];
        alt[x + 1][y] = arr[5];
        alt[x + 1][y + 1] = arr[2];
    }
}

cout << "Output:" << endl;
for (int ii = 0; ii < 5; ii++)
{
    for (int jj = 0; jj < 4; jj++)
    {
        cout.width(3);
        cout.setf(ios::left);
        cout << alt[ii][jj];
    }
    cout << alt[ii][4] << endl;
}

return 0;
}

```

P706

DIFFICUTLY ★★★★★

输入任意 10 个整数，对这 10 个整数从小到大排序并输出。

可用素材： printf("Please input 10 number:\n")
printf("%5d"...

程序的运行效果应类似地如图 706.1 所示，金色部分是键盘输入的内容。

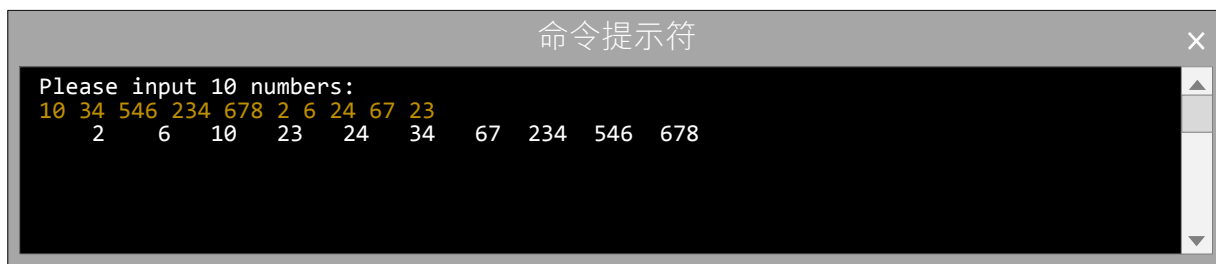


图 706.1

参考答案:

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

```
int main(void)
{
    int arr[10], i, j, tmp;

    printf("Please input 10 number:\n");
    for (i = 0; i < 10; i++)
    {
        scanf("%d", &arr[i]);
    }

    for (j = 1; j < 10; j++)
    {
        for (i = 0; i < 10 - j; i++)
        {
            if (arr[i] > arr[i + 1])
            {
                tmp = arr[i];
                arr[i] = arr[i + 1];
                arr[i + 1] = tmp;
            }
        }
    }

    for (i = 0; i < 10; i++)
    {
        printf("%5d", arr[i]);
    }
    putchar('\n');

    return 0;
}
```

P774

DIFFICUTLY ★★★★★

程序 P774.c 已编写部分代码（见文件内容），请根据程序中的要求完善程序（在指定的位置添加代码或将 `_____` 换成代码）。

程序的功能是：

1. 创建一个链表，共有 N (N 由 `#define` 定义) 个节点，第 1 个节点的数据域赋值为 0，第 2 个节点的数据域赋值为 1，以此类推，第 20 个节点的数据域赋值为 19。
2. 输出该链表节点的数据，如图 774.1 所示（“before” 所在行）。
3. 通过改变指针域的指向，将链表倒序，并输出倒序后的链表各节点数据，如图 774.1 所示（after 所在行）。

程序的运行效果应类似地如图 774.1 所示。



图 774.1

附件: 
P774.c

文件内容:

P774.c

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

#define N 20

/* User Code Begin(Limit: lines<=5, lineLen<=60, 考生可在本行后添加代码、仅5行、行长<=60字符, 功能是节点类型Link定义) */
<A>
/* User Code End(考生添加代码结束) */

void dispLink(struct Link *Head); /* 根据给定的链首Head, 显示整个链表中的数据, 同时检查两次使用的链表是否刚好反序 */
/* User Code Begin(Limit: lines<=2, lineLen<=60, 考生可在本行后添加代码、仅2行、行长<=60字符, 功能是用户自定义函数的原型声明) */
<B>
/* User Code End(考生添加代码结束) */

int main(void)
{
    struct Link *Head;

    Head = creatLink(); /* 创建新链表 */
    printf("\nbefore:");
    dispLink(Head);

    Head = reverseLink(Head); /* 反转链表 */
    printf("\nafter:");
    dispLink(Head);

    return 0;
}

void dispLink(struct Link *Head)
{
    static struct Link *oLink[2][N];
    static int callNumber = -1;
    int i = 0;

    callNumber++;
    if (callNumber > 1)
    {
```

```

        printf("Error, call invalid!\n");
        return ;
    }

    while (Head != NULL)
    {
        oLink[callNumber][i] = Head;
        i++;
        printf("%d ", Head->data);
        Head = Head->next;
    }
    printf("\n");

    if (1 == callNumber)
    {
        for (i=0; i<N; i++)
        {
            if (oLink[0][i] != oLink[1][N-1-i])
            {
                printf("Error, Link not reverse!\n");
                return ;
            }
        }
    }
}

/* User Code Begin(考生在此后根据设计需要完成程序的其它部分, 行数不限) */
<C>

```

参考答案:

```

<A> struct Link
{
    int data;
    struct Link *next;
};
<B> struct Link *creatLink(void);
    struct Link *reverseLink(struct Link *Head);
<C> struct Link *creatLink(void)
{
    struct Link *Head, *p1, *p2;
    int i;

    Head = p1 = p2 = (struct Link *)malloc(sizeof(struct Link));
    for (i = 0; i < N; i++)
    {
        p1->data = i;
        if (Head == NULL)
        {
            Head = p1;
        }
        else
        {
            p2->next = p1;
        }
        p2 = p1;
        p1 = (struct Link *)malloc(sizeof(struct Link));
    }
    p2->next = NULL;

    return Head;
}

```

```

}

struct Link *reverseLink(struct Link *Head)
{
    struct Link *h, *p1, *p2;
    int flag = 0;

    h = p1 = p2 = Head;
    while (h != NULL)
    {
        if (flag == 0)
        {
            h = h->next;
            p1->next = NULL;
            flag++;
        }
        else if (flag == 1)
        {
            p1 = h;
            h = h->next;
            p1->next = p2;
            flag++;
        }
        else
        {
            p2 = p1;
            p1 = h;
            h = h->next;
            p1->next = p2;
        }
    }

    return p1;
}

```

Appendix A C Operators

优先级	运算符	名称或含义	使用形式	结合方向	运算类
1 (High)	()	圆括号	(表达式)/函数名(形参表)	→ (从左到右)	
	[]	数组下标	数组名[常量表达式]		
	->	成员选择(指针)	对象指针->成员名		
	.	成员选择(对象)	对象.成员名		
2	!	逻辑非运算符	!表达式	← (从右到左)	单目运算
	~	按位取反运算符	~表达式		
	++	自增运算符	++变量名/变量名++		
	--	自减运算符	--变量名/变量名--		
	-	负号运算符	-表达式		
	(类型)	强制类型转换运算符	(数据类型)表达式		
	*	取值运算符	*指针变量		
	&	取地址运算符	&变量名		
3	sizeof	长度运算符	sizeof(表达式)	→	双目算术运算
	*	乘法运算符	表达式 * 表达式		
	/	除法运算符	表达式 / 表达式		
4	%	余数(取模)运算符	整型表达式 % 整型表达式	→	双目算术运算
	+	加法运算符	表达式 + 表达式		
	-	减法运算符	表达式 - 表达式		
5	<<	左移运算符	变量 << 表达式	→	移位运算
	>>	右移运算符	变量 >> 表达式		
6	<	小于运算符	表达式 < 表达式	→	关系运算
	<=	小于等于运算符	表达式 <= 表达式		
	>	大于运算符	表达式 > 表达式		
	>=	大于等于运算符	表达式 >= 表达式		
7	==	等于运算符	表达式 == 表达式	→	关系运算
	!=	不等于运算符	表达式 != 表达式		
8	&	按位与运算符	表达式 & 表达式	→	位运算
9	^	按位异或运算符	表达式 ^ 表达式	→	位运算
10		按位或运算符	表达式 表达式	→	位运算
11	&&	逻辑与运算符	表达式 && 表达式	→	逻辑运算
12		逻辑或运算符	表达式 表达式	→	逻辑运算
13	?:	条件运算符	表达式1 ? 表达式2 : 表达式3	←	三目运算
14	=	赋值运算符	变量 = 表达式	←	双目运算
	/=	除后赋值运算符	变量 /= 表达式		
	*=	乘后赋值运算符	变量 *= 表达式		
	%=	取模后赋值运算符	变量 %= 表达式		
	+=	加后赋值运算符	变量 += 表达式		
	-=	减后赋值运算符	变量 -= 表达式		
	<<=	左移后赋值运算符	变量 <<= 表达式		
	>>=	右移后赋值运算符	变量 >>= 表达式		
	&=	按位与后赋值运算符	变量 &= 表达式		
	^=	按位异或后赋值运算符	变量 ^= 表达式		
	=	按位或后赋值运算符	变量 = 表达式		
15 (Low)	,	逗号运算符	表达式1, 表达式2, ...	→	顺序运算

Appendix B ASCII Charmap

值	缩写/字符	解释	值	字符	值	字符	值	缩写/字符	值	字符	值	字符	值	字符	值	字符
000	NUL (null)	空字符	032	(space)	064	@	096	`	128	Ç	160	á	192	ˆ	224	α
001	☺ SOH (start of headline)	标题开始	033	!	065	A	097	a	129	ü	161	í	193	⌞	225	β
002	☺ STX (start of text)	正文开始	034	"	066	B	098	b	130	é	162	ó	194	⌞	226	Γ
003	♥ ETX (end of text)	正文结束	035	#	067	C	099	c	131	â	163	ú	195	⌞	227	π
004	♦ EOT (end of transmission)	传输结束	036	\$	068	D	100	d	132	ä	164	ñ	196	—	228	Σ
005	♣ ENQ (enquiry)	请求	037	%	069	E	101	e	133	à	165	Ñ	197	†	229	σ
006	♠ ACK (acknowledge)	收到通知	038	&	070	F	102	f	134	å	166	æ	198	‡	230	μ
007	• BEL (bell)	响铃	039	'	071	G	103	g	135	ç	167	º	199	‡	231	τ
008	␣ BS (backspace)	退格	040	(072	H	104	h	136	ê	168	¿	200	⌞	232	Φ
009	○ HT (horizontal tab)	水平制表符	041)	073	I	105	i	137	ë	169	ˆ	201	⌞	233	Θ
010	␣ LF (NL line feed, new line)	换行键	042	*	074	J	106	j	138	è	170	ˆ	202	⌞	234	Ω
011	♂ VT (vertical tab)	垂直制表符	043	+	075	K	107	k	139	ï	171	½	203	⌞	235	δ
012	♀ FF (NP form feed, new page)	换页键	044	,	076	L	108	l	140	î	172	¾	204	⌞	236	∞
013	♪ CR (carriage return)	回车键	045	-	077	M	109	m	141	ì	173	ı	205	=	237	φ
014	♪ SO (shift out)	不用切换	046	.	078	N	110	n	142	Ä	174	«	206	⌞	238	ε
015	✱ SI (shift in)	启用切换	047	/	079	O	111	o	143	Å	175	»	207	⌞	239	η
016	► DLE (data link escape)	数据链路转义	048	0	080	P	112	p	144	É	176	␣	208	⌞	240	≡
017	◄ DC1 (device control 1)	设备控制 1	049	1	081	Q	113	q	145	æ	177	␣	209	⌞	241	±
018	↕ DC2 (device control 2)	设备控制 2	050	2	082	R	114	r	146	Æ	178	␣	210	⌞	242	≥
019	!! DC3 (device control 3)	设备控制 3	051	3	083	S	115	s	147	ô	179	␣	211	⌞	243	≤
020	Ⓜ DC4 (device control 4)	设备控制 4	052	4	084	T	116	t	148	ö	180	␣	212	Ô	244	⌈
021	§ NAK (negative acknowledge)	拒绝接收	053	5	085	U	117	u	149	ò	181	␣	213	⌞	245	⌋
022	▬ SYN (synchronous idle)	同步空闲	054	6	086	V	118	v	150	û	182	␣	214	⌞	246	÷
023	‡ ETB (end of trans. block)	传输块结束	055	7	087	W	119	w	151	ù	183	␣	215	⌞	247	≈
024	↑ CAN (cancel)	取消	056	8	088	X	120	x	152	ÿ	184	␣	216	⌞	248	≈
025	↓ EM (end of medium)	介质中断	057	9	089	Y	121	y	153	Ö	185	␣	217	⌞	249	•
026	→ SUB (substitute)	替补	058	:	090	Z	122	z	154	Ü	186	␣	218	⌞	250	•
027	← ESC (escape)	换码 (溢出)	059	;	091	[123	{	155	¢	187	␣	219	␣	251	√
028	⌞ FS (file separator)	文件分割符	060	<	092	\	124		156	£	188	␣	220	␣	252	ˆ
029	↔ GS (group separator)	分组符	061	=	093]	125	}	157	¥	189	␣	221	␣	253	ˆ
030	▲ RS (record separator)	记录分离符	062	>	094	^	126	~	158	℞	190	␣	222	␣	254	■
031	▼ US (unit separator)	单元分隔符	063	?	095	_	127	␣ DEL (delete)	159	f	191	␣	223	␣	255	

标准 ASCII 码表 (0 ~ 127)

拓展 ASCII 码表 (128 ~ 255)

Appendix C Data Types

数据类型所占字节因编译器的不同而不同，可用 `sizeof(数据类型)` 查看。

数据类型		字节	取值范围及备注	
int	有符号基本整型	2	-32768 ~ 32767	$-2^{15} \sim (2^{15} - 1)$
		4*	-2147483648 ~ 2147483647	$-2^{31} \sim (2^{31} - 1)$
unsigned int	无符号基本整型	2	0 ~ 65535	$0 \sim (2^{16} - 1)$
		4*	0 ~ 4294967295	$0 \sim (2^{32} - 1)$
short	有符号短整型	2	-32768 ~ 32767	$-2^{15} \sim (2^{15} - 1)$
unsigned short	无符号短整型	2	0 ~ 65535	$0 \sim (2^{16} - 1)$
long	有符号长整型	4	-2147483648 ~ 2147483647	$-2^{31} \sim (2^{31} - 1)$
unsigned long	无符号长整型	4	0 ~ 4294967295	$0 \sim (2^{32} - 1)$
long long**	有符号双长整型	8	-9223372036854775808 ~ 9223372036854775807	$-2^{63} \sim (2^{63} - 1)$
unsigned long long**	无符号双长整型	8	0 ~ 18446744073709551615	$0 \sim (2^{64} - 1)$
char	有符号字符型	1	-128 ~ 127	$-2^7 \sim (2^7 - 1)$
unsigned char	无符号字符型	1	0 ~ 255	$0 \sim (2^8 - 1)$
float	单精度浮点型	4	绝对值: 0 以及 $1.2 \times 10^{-38} \sim 3.4 \times 10^{38}$	有效数字: 6
double	双精度浮点型	8	绝对值: 0 以及 $2.3 \times 10^{-308} \sim 1.7 \times 10^{308}$	有效数字: 15
long double	长双精度浮点型	8*	绝对值: 0 以及 $2.3 \times 10^{-308} \sim 1.7 \times 10^{308}$	有效数字: 15
		16	绝对值: 0 以及 $3.4 \times 10^{-4932} \sim 1.1 \times 10^{4932}$	有效数字: 19

* 该类型在 Visual C++/Visual Studio 所占字节。

** C99 标准新增类型。

Appendix D C Keywords

C11 标准的 C 关键字。不允许把关键字用作变量名称。

auto	if	unsigned
break	inline	void
case	int	volatile
char	long	while
const	register	_Alignas
continue	restrict	_Alignof
default	return	_Atomic
do	short	_Bool
double	signed	_Complex
else	sizeof	_Generic
enum	static	_Imaginary
extern	struct	_Noreturn
float	switch	_Static_assert
for	typedef	_Thread_local
goto	union	

Appendix E Exam

共 10 题，满分 100 分，总分 126 分。考试用时 170 分钟。

2014.01.15

1 (7 分 | 数据类型、输入/输出 | 新题)

先从键盘读入 7 个数据（依次为 3 整数、2 字符、2 实数），然后按示例格式倒序输出这 7 个数据。

可用素材： `printf("请输入7个数据(依次为3整数、2字符、2实数): ")`

`printf("\n这7个数据倒序为: ")`

程序的运行效果应类似地如图 E-1.1 所示，金色部分是从键盘输入的内容。

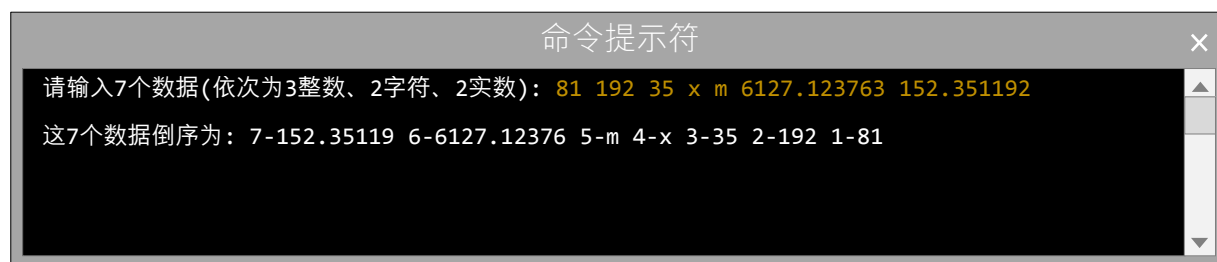


图 E-1.1

参考答案：

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

```
int main(void)
```

```
{
```

```
    int num1, num2, num3;
```

```
    char ch1, ch2;
```

```
    double num4, num5;
```

```
    printf("请输入7个数(依次为3整数、2字符、2实数): ");
```

```
    scanf("%d%d%d %c %c%lf%lf", &num1, &num2, &num3, &ch1, &ch2, &num4, &num5);
```

```
    printf("\n这7个数倒序为: ");
```

```
    printf("7-%.5f 6-%.5f 5-%c 4-%c 3-%d 2-%d 1-%d\n", num5, num4, ch2, ch1, num3, num2, num1);
```

```
    return 0;
```

```
}
```

2 (10 分 | 选择结构、表达式 | 新题)

从键盘读入 3 个整数，按由小到大的顺序输出这 3 个数。

可用素材： `printf("请输入3个数: ")`

`printf("\n这三个数由小到大为: %d %d %d\n"...`

程序的运行效果应类似地如图 E-2.1 所示，金色部分是从键盘输入的内容。

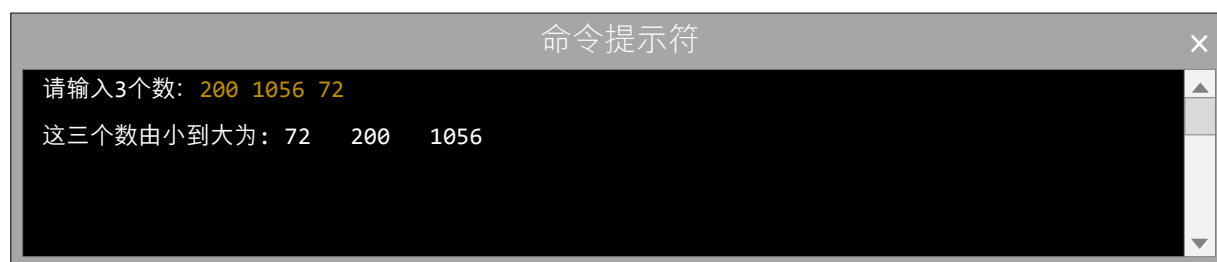


图 E-2.1

参考答案：

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



```

int main(void)
{
    int num1, num2, num3, tmp;

    printf("请输入3个数: ");
    scanf("%d%d%d", &num1, &num2, &num3);
    if (num1 > num2)
    {
        tmp = num1;
        num1 = num2;
        num2 = tmp;
    }
    if (num1 > num3)
    {
        tmp = num1;
        num1 = num3;
        num3 = tmp;
    }
    if (num2 > num3)
    {
        tmp = num2;
        num2 = num3;
        num3 = tmp;
    }
    printf("\n这三个数由小到大为: %d  %d  %d\n", num1, num2, num3);

    return 0;
}

```

3 (10 分 | 命令行、选择结构 | 参见原题, 如 P118)

4 (11 分 | 循环结构、输入/输出 | 改编自 P125)

从键盘读入两个字符 *cBegin* 和 *cEnd*, 要求按示例格式依次输出 $\leq cBegin$ 且 $\geq cEnd$ 的每一字符及其下一字符。

可用素材: `printf("Please Input two char: ")`
`printf("\nResult: ")`

程序的运行效果应类似地如图 E-4.1 所示, 金色部分是从键盘输入的内容。



图 E-4.1

参考答案:

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

```

int main(void)
{
    char cBegin, cEnd;
    int i;

    printf("Please Input two char: ");
    cEnd = getchar();
    cBegin = getchar();
}

```

```

printf("\nResult: ");
for (i = cEnd; i >= cBegin; i--)
{
    printf("%c%c", i, i + 1);
}
putchar('\n');

return 0;
}

```

5 (12 分 | 字符串、表达式 | 改编自 P220)

从键盘读入一行字符（约定：字符数 ≤ 127 字节），统计该行字符中出现的字母、数字和其它字符的数量，最后按示例格式输出这三类字符的数量及其内容。

可用素材：

```

printf("Please input the string: ")
printf("\nDigistal string length = ..., Digital: ...\n"...
printf("Letter string length = ..., Letters: ...\n"...
printf("Symbol string length = ..., Symbols: ...\n"...

```

程序的运行效果应类似地如图 E-5.1 所示，金色部分是从键盘输入的内容。



图 E-5.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    char str[128], dig[128], let[128], sym[128];
    int i, count1 = 0, count2 = 0, count3 = 0;

    printf("Please input the string: ");
    gets(str);

    for (i = 0; str[i] != '\0'; i++)
    {
        if (str[i] >= '0' && str[i] <= '9')
        {
            dig[count1] = str[i];
            count1++;
        }
        else if (str[i] >= 'A' && str[i] <= 'Z' || str[i] >= 'a' && str[i] <= 'z')
        {
            let[count2] = str[i];
            count2++;
        }
        else
        {
            sym[count3] = str[i];
            count3++;
        }
    }
}

```

```

    dig[count1] = let[count2] = sym[count3] = '\0';
    printf("\nDigital string length = %d, Digitals: %s\n", count1, dig);
    printf("Letter string length = %d, Letters: %s\n", count2, let);
    printf("Symbol string length = %d, Symbols: %s\n", count3, sym);

    return 0;
}

```

6 (13 分 | 数组、输入/输出 | 改编自 P818、P313)

先从键盘读入整数 m 和 n (约定 $2 \leq m \leq 20$, $2 \leq n \leq 20$)，再从键盘读入 m 行 (每行 n 个，即一个 $m \times n$ 矩阵) 整数，然后从键盘读入一个列序号，按示例格式显示该列的内容。

可用素材：

```

printf("请输入 m 和 n: ")
printf("请输入 %d 行，每行 %d 列整数:\n"...
printf("请输入要显示列的列号: ")
printf("\n该列的内容为: ")

```

程序的运行效果应类似地如图 E-6.1 所示，金色部分是从键盘输入的内容。



图 E-6.1

参考答案：

```

#include <stdio.h>

int main(void)
{
    int m0, n0, arr[20][20], i, j, dis;

    do
    {
        printf("请输入 m 和 n: ");
        scanf("%d%d", &m0, &n0);
    } while (m0 < 2 || m0 > 20 || n0 < 2 || n0 > 20);
    printf("请输入 %d 行，每行 %d 列整数:\n", m0, n0);
    for (i = 0; i < m0; i++)
    {
        for (j = 0; j < n0; j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }

    do
    {
        printf("请输入要显示列的列号: ");
        scanf("%d", &dis);
    } while (dis < 0 || dis > n0);
    printf("\n该列的内容为: ");
    for (i = 0; i < m0; i++)

```

```

    {
        printf("%d ", arr[i][dis]);
    }
    putchar('\n');

    return 0;
}

```

7 (13 分 | 函数基本应用、查找算法 | 参见原题, 如 P246)

8 (14 分 | 函数高级应用、指针 | 改编自 P240)

程序 E-P8.c 已编写部分代码 (见文件内容), 请根据程序中的要求完善程序 (在指定的位置添加代码或将 换成代码)。

程序的功能是: 分 3 次调用自定义函数实现从键盘为数组 **arr₁**、**arr₂**、**arr₃** 分别读入 6、10、15 个数并计算每一组数的和及其中大于 60 的数的个数, 然后分别输出每一数组头尾两个元素的值及所有元素的和、大于 60 的数的个数。

可用素材: printf("请输入%d个数: "...

程序的运行效果应类似地如图 E-8.1 所示, 金色部分是从键盘输入的内容。

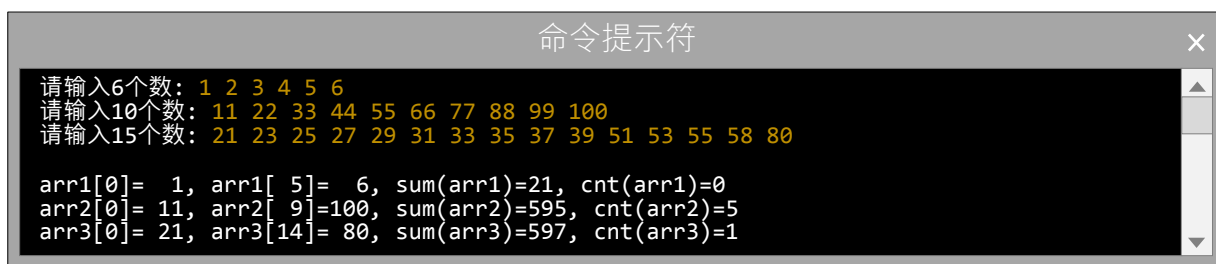



图 E-8.1

附件:  E-P8.c

文件内容:

E-P8.c

```

#include <stdio.h>

/* userCode(<60字符>): 自定义函数之原型声明 */
<A> _____

int main(void)
{
    int arr1[6], arr2[10], arr3[15], gt60A, gt60B, gt60C, sumA, sumB, sumC;

    <B> _____ /* userCode(<60字符>): 调用函数读 6个数到arr1中, 并计算和及>60的个数 */
    <C> _____ /* userCode(<60字符>): 调用函数读10个数到arr2中, 并计算和及>60的个数 */
    <D> _____ /* userCode(<60字符>): 调用函数读15个数到arr3中, 并计算和及>60的个数 */

    printf("\narr1[0]=%3d, arr1[ 5]=%3d, sum(arr1)=%d, cnt(arr1)=%d", arr1[0], arr1[5], sumA,
gt60A);
    printf("\narr2[0]=%3d, arr2[ 9]=%3d, sum(arr2)=%d, cnt(arr2)=%d", arr2[0], arr2[9], sumB,
gt60B);
    printf("\narr3[0]=%3d, arr3[14]=%3d, sum(arr3)=%d, cnt(arr3)=%d\n", arr3[0], arr3[14],
sumC, gt60C);

    return 0;
}

/* User Code Begin: 考生在此后完成自定义函数的设计, 行数不限 */

```

<E>

参考答案:

```
<A> int Sum(int arr[], int n, int *pgt60);
<B> sumA = Sum(arr1, 6, &gt60A);
<C> sumB = Sum(arr2, 10, &gt60B);
<D> sumC = Sum(arr3, 15, &gt60C);
<E> int Sum(int arr[], int n, int *pgt60)
{
    int i, sum = 0, count = 0;

    printf("请输入%d个数: ", n);
    for (i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
        sum += arr[i];
        if (arr[i] > 60)
        {
            count++;
        }
    }
    *pgt60 = count;

    return sum;
}
```

9 (16 分 | 文件操作、综合 | 参见原题, 如 P324)

10 (20 分 | 排序算法、综合 | 参见原题, 如 P419)