# 张引老师班 2020-2021 秋冬 C 语言期中测试 2 详解

## 目录



## 写在最前面（错误检查表）

有同学老问，老师，段错误是什么！我不知道怎么调！
段错误一共有两种：
1. scanf 没有加地址符号（**最常见**！另外，%s 千万别加地址符！）
scanf(“%d”, &time.minute);
另外，注意 scanf 里面千万不能加回车，加了有 80%几率答案错误！

2. 数组下标越界
下标越界也有两种情况：
2.1 开的数组不够大！ 题目中告你 n <100，你就开 1000 准没错！但是记住数组各维度合计大小不要超过一千万！
2.2 访问错误的下标！一般是你有一个存最大数字的下标 min，你没有初始化它，导致它是一个随机值，你访问 a[min]当然下标越界了！

另外，还有其他一些错误：
格式错误： 你的**空格数量**和答案不一样！人家让你输出几位，中间空及格？用什么样的格式化输出控制？？

编译错误：
1. error: stray‘\357’ in program 或者 stray ‘\273’ in program
错误的原因是你写了中文的分号/括号/逗号/句号！！自己回去看看，检查不出来就全删除了自己再打一次。
2. permission denied: exit with 1 status
错误的原因是你编译运行前没有关闭上次的程序！！
3.另外，struct 定义的**最后面没有加分号**，也会导致编译错误！！（看不懂我这句话的回去复习结构！）

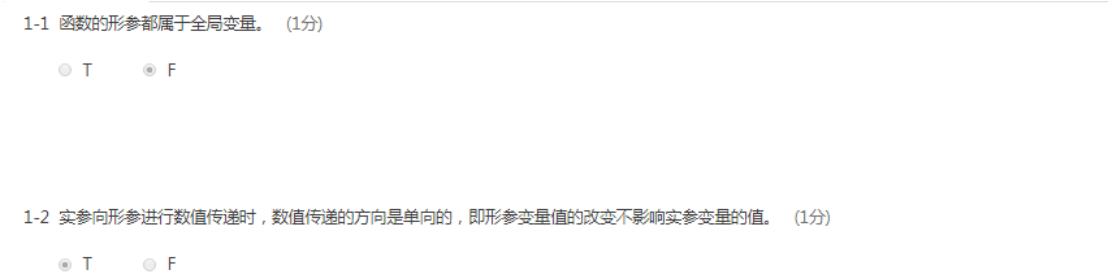
运行超时：
1. 多半是**死循环**！也就是你逻辑有问题！
2. 是否有多余输入？？题目给你两个数，你输入三个数，系统等不到就会超时！
3. 是否是素数题？如果是，改成根号 N 的算法！
4. 是否是要计算阶乘等超大数字？如果是，改成单步计算（具体见期中测试 1 详解的编程题第一题！）

答案错误：
如果你错了一两个点：
1. 和人家的字母大小写是否一样？
2. 边界情况如 0,1,2,最大，特殊情况如 None 是否处理了？

如果你一个都没对：
赶紧对照测试区域的输出和标准输出，看看到底差在哪里了！
如果根本看不出来差在哪里，全都是乱码：
肯定是初始化有问题！你赋值初值了嘛？初值应该在循环内部还是外部赋值？？
如果不是乱码，看上去挺规则的但是全错了：
肯定是算法设计的有问题！先**想想**自己的算法有没有明显的漏洞，如果想不出来就输出中间结果！
怎么输出中间结果？？
如果是数组题，算法每运行一步就输出整个数组看看（记得中间打空格行末打回车）
如果没有数组，算法每运行一步就输出你的重要变量看看
如果有子程序（函数）,先不要管题目，带入几个特殊值，检查你的函数是否正常工作！

注意，输出中间结果是想不出来实在无计可施才用的方法，能想就想，能节省很多时间

# 判断题



1-3 下面这段代码不规范的，程序可能运行出错，原因是：通过指针p访问局部变量tmp，而tmp的空间在函数运行结束后已经释放。 (1分)

```
int* hoho(int n){
    static int tmp;
    tmp += n;
    return &tmp;
}
/* 此处省略若干行代码 */
int main(){
    /* 此处省略若干行代码 */
    int *p;
    p = hoho(3);
    /* 此处省略若干行代码，且这些代码不会修改p的值 */
    *p = 6;
    return 0;
}
```

○ T ◉ F

静态变量不会在函数运行结束后就释放，会一直保存。

1-4 执行以下程序段，sum的值是1.5。 (1分)

```
int i, sum;
sum = 0;
for (i = 1; i <= 2; i++){
    sum = sum + 1.0/i;
}
```

○ T ◉ F

---

看好了，sum 是 int 类型！这种陷阱题是期末**最喜欢**考的！

1-5 对于如下的switch语句（使用break）的一般形式，其执行流程是：首先求解表达式，如果表达式的值与某个常量表达式的值相等，则执行该常量表达式后的相应语句段；如果表达式的值与任何一个常量表达式的值都不相等，则执行 default 后的语句段，最后执行breaki语句，跳出switch语句。 (1分)

```
switch (表达式) {
    case 常量表达式1: 语句段1; break;
    case 常量表达式2: 语句段2; break;
              …
    case 常量表达式n: 语句段n; break;
    default:          语句段n+1; break;
}
```

◉ T ○ F

---

1-6 若变量已正确定义，执行以下程序段，输入负数时，循环结束。 (1分)

```
total = 0;
scanf ("%d", &score);
while(score >= 0){
    total = total + score;
    scanf ("%d", &score);
}
```

◉ T ○ F

1-7 `sizeof(int)` 可计算整型所占的内存字节数，但是 `sizeof( )` 并不是一个函数，而是一个运算符（操作符，operator）。 (1分)

◉ T ○ F

1-8 函数一次可以返回两个以上的值。 (1分)

○ T ◉ F

不可以，这里指的是 return 后面只能跟最多一个值。

1-9 关于C语言指针的运算：指针只有加减操作，没有乘除操作。指针可以加常数、减常数；相同类型的指针可以相加、相减。 (1分)

○ T ◉ F

指针没有加、乘、除，只有同类可以减。

1-10 结构体变量可以作数组元素。 (1分)

◉ T ○ F

1-11 数组的基地址是在内存中存储数组的起始位置，数组名本身就是一个地址即指针值。 (1分)

◉ T ○ F

1-12 调用strcmp函数比较字符串大小时，通常较长的字符串会较大。 (1分)

○ T ◉ F

这个题是期中测试 1 原题。

1-13 不同类型的结构变量之间也可以直接赋值。 (1分)

○ T ◉ F

1-14 假设 `k` 是整型变量，计算表达式 `1.0/k` 后结果的数据类型是浮点型。 (1分)

◉ T ○ F

是的，有 1.0 参与计算就会变为 double。注意这个题和 1-4 的区别，本题并没有让 k=1.0/k。

1-15 `double a = 5E-3;` 是非法语句。 (1分)

○ T ◉ F

5E-3 是科学计数法，相当于 5*10^-3.

# 单选题

2-7 运行以下程序后，如果从键盘上输入65 14<回车>，则输出结果为（ ）。 (2分)

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

      printf("Enter m,n;");
      scanf("%d%d", &m,&n);
      while (m != n)
      {   while (m > n)       m = m - n;
          while (n > m)       n = n - m;
      }
      printf("m=%d\n",m);

      return 0;
}
```

- A. m=3
- B. m=2
- C. m=1
- D. m=0

这个题也是更相减损术，在期中 1 里讲过了。

2-9 有说明语句 `int a[4][5];` ，则 `a[2]+3` 表示__。 (2分)

- A. `a` 数组行下标为 `2` 、列下标为 `3` 的元素的地址
- B. `a` 数组行下标为 `2` 、列下标为 `3` 的元素的值
- C. `a` 数组第 `2` 行的首地址
- D. `a` 数组第 `3` 行的首地址

a[2] 是下标为 2 的行，这一行也是一个一维数组。

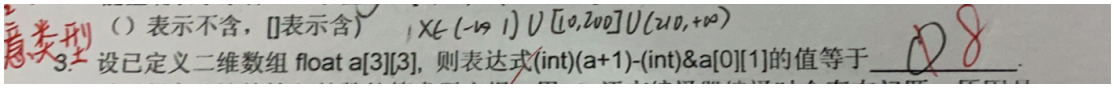
大家可以观察以下等式：
*(a[2] + 3) = a[2][3]
(*(a+2))[3] = a[2][3]

*(*(a+2)+3) = a[2][3]

&a[2][3] = a[2] + 3
&(&a[2][3]-3) = a + 2

首先，若 a 是数组或指针，x 是整数，*(a+x)运算和 a[x]运算是等价的。
这里我们可以发现，a+2 指向的是下标为 2 的行的首地址。
对于这个二维数组 a 来说，a+2 和 a 之间相差了 5*2 = 10 个 int 的位置。

再看下面一个题：




在这个题中，a+1 指的是行号为 1 的行首，a[0][1]指的是第 0 行的第 1 个元素，
他们之间相差了两个 float 的位置。将地址转为整数再相减，实际上就是相差了 8 个字节(一个 float 是四个字节)。

没有 2-10

2-11 *字符数组(1-2) (2分)

下面的程序段将输出 _____。

```
char s[10] = "abcd";
printf("%d\n", sizeof(s));
```

- A. 4
- B. 5
- C. 10
- D. 11

sizeof()是数组大小，就是 char s[10] 的 10.

2-12 *二维数组(2-3) (2分)

下面的程序段将输出 _____。

```
double a[3][3] = {1, 2, 3, 4, 5
printf("%g\n", a[2][2]);
```

- A. 0
- B. 1
- C. 2
- D. 3
- E. 4
- F. 5

老师观点：如果是全局的，是 0，如果是局部的，不确定，这道题不严谨
揣摩出题人意图（仅代表助教个人观点）：如果选项里出现了“不确定”的话，他考你的就是局部变量随机初始化的特点。如果没有，那统一按 0 算吧。

2-13 Which statement can initialize a one-dimensional array? (2分)

- A. int a[10]={0,0,0,0,0};
- B. int a[2]={0,1,2};
- C. int a[]={0};
- D. int a[]={};

A.是圆括号，不行
B.下标越界了
C.对了
D.不能初始化为空

2-14 在执行 int a[][3]={1,2,3,4,5,6};语句后， a[1][0]的值是（ ）。 (2分)

- A. 4
- B. 1
- C. 2
- D. 5

2-15 以下程序的运行结果是__ 。 (2分)

```
#include <stdio.h>
int main()
{
int p[7]={11,13,14,15,16,17,18},i=0,k=0;
while(i<7&&p[i]%2)
{
   k=k+p[i];
   i++;
}
printf("%d\n",k);
return 0;
}
```

- A. 58
- B. 56
- C. 45
- D. 24

这个题是将 p 数组中奇数加起来（就是 p[i] % 2 == 1），
他是一个 while 循环，因此遇到第一个偶数就会停下。
因此是 11+13 = 24

2-46 如果一个变量在整个程序运行期间都存在，但是仅在说明它的函数内是可见的，这个变量的存储类型应该被说明为( )。 (2分)

- A. 静态变量
- B. 动态变量
- C. 外部变量
- D. 内部变量

2-47 执行下面程序，正确的输出是（ ）。 (2分)

```
int x = 5, y = 7;
void swap ( )
{
      int z ;

      z = x ;  x = y ;  y = z ;
}
int main(void)
{
      int x = 3, y = 8;
      swap ( ) ;
      printf ( " %d , %d \n", x , y ) ;

      return 0 ;
}
```

- A. 3 , 8
- B. 8 , 3
- C. 5 , 7
- D. 7 , 5

注意：
1. 此处一共有全局 x,y 和 main（ ）函数里的 x,y，他们是不一样的。
2. 在 main 函数中，局部的 x,y 生效，会隐藏全局的 x,y。printf()输出的也是局部的 x,y。
3. swap()函数中没有同名的局部变量，因此交换的是全局的 x,y。

2-48 5-5. 下列程序的输出结果是 (2分)

```
#include <stdio.h>

int fun3(int x) {
  static int a = 3;
  a += x;
  return (a);
}

int main()
{
  int k = 2, m = 1, n;
  n = fun3(k); n = fun3(m);
  printf("%d\n", n);
  return 0;
}
```

- A. 3
- B. 4
- C. 6
- D. 9

静态变量。这种题在期中 1 详解里详细讲过了，不再赘述。

2-49 文件中定义的全局变量的作用域为（ ）。 (2分)

- A. 本程序全部范围
- B. 本文件全部范围
- C. 函数内全部范围
- D. 从定义该变量的位置开始到本文件结束

2-50 执行以下程序，打印输出的内容是： (2分)

```
#include <stdio.h>
int x=5, y=6;
void incxy( ){
    x++; y++;
}
int main( ){
    int x=3;
    incxy( );
    printf("%d,%d\n", x,y);
    return 0;
}
```

- A. 3,6
- B. 4,7
- C. 3,7
- D. 6,7

同样考察的是全局变量。

2-51 若a、b、c1、c2、x、y均是整型变量，以下正确的switch语句是 (2分)

A.
```
switch(a+b);
{    case 1: y=a+b; break;
     case 0:y=a-b; break;}
```

B.
```
switch(a*a+b*b)
{
case 3:
case 1: y=a+b; break;
case 3: y=b-a, break;}
```

C.
```
switch a
{    case c1: y=a-b; break;
case c2: x=a*b; break;
default: x=a+b; }
```

D.
```
switch(a-b)
{    default: y=a*b; break;
case 3: case 4: x=a+b; break;
case 10: case 11: y=a-b; break; }
```

---

A. switch 后面不能直接加分号

B. case 标签不能重复

C. case 标签不能是变量。（实际上，应该只能是整数**常量如 1,2,3** 或者整数**常量**表达式如 1+1*3+3 ）,至于 a 外面是否需要括号，建议大家自己尝试。

2-52 请读程序: (2分)

```
#include<cstdio>
using namespace std;

int main()
{    float x,y;

     scanf("%f",&x);

     if(x<0.0) y=0.0;
     else if((x<5.0)&&(x!=2.0))
        y=1.0/(x+2.0);
     else if (x<10.0) y=1.0/x;
     else y=10.0;
     printf("%f\n",y);

     return 0;

}
```

若运行时从键盘上输入2.0(表示回车)，则上面程序的输出结果是（ ）

- A. 0.000000
- B. 0.250000
- C. 0.500000
- D. 1.00000

2-53 请阅读以下程序段： (2分)

```
int a=5, b=0, c=0;
if ( a=b+c )
  printf("Equal!\n");
else
  printf("Not Equal!\n");
```

以上程序（ ）。

- A. 有语法错误不能通过编译
- B. 可以通过编译但是不能通过连接
- C. 输出 Not Equal !
- D. 输出 Equal !

if(a=b+c) 这里是赋值语句，虽然逻辑可能有问题，但编译不会出错。这里就是强制让 a 赋值为 b+c 的值，然后以这个值返回 if()的判断，即 0.

# 填空题

4-1 以下程序运行后的输出结果是 8778 (3分)

```
#include<stdio.h>
int main( )
{
    int x=7,y=8,*p,*q,*r;
    p = &x; q = &y;
    r = p; p = q; q = r;
    printf("%d%d%d%d\n",*p,*q, x, y);
    return 0;
}
```

交换指针不交换值。

4-3 下面程序的输出结果是 bcdefgha (2分)

```
#include <stdio.h>
char* ss(char *s) {
    char *p, t;
    p = s + 1; t = *s;
    while (*p) {
        *(p - 1) = *p;
        p++;
    }
    *(p - 1) = t;
    return s;
}
int main()
{
    char *p, str[10] = "abcdefgh";
    p = ss(str);
    printf("%s\n", p);
    return 0;
}
```

这里做的是循环左移一位，大家可以记一下。分为三步：

1. 记录首位的值
2. 一个循环，每个值向前移动一位
3. 将记录下的值填到末尾。 如果这个题是程序填空，你会不会？非常容易考出来。

4-4 以下程序运行后输入“3,abcde<回车>”，则输出结果是 cdeab (2分)。

```
#include <stdio.h>
#include <string.h>
void move(char* str, int n) {
    char temp;
    int i;
    temp = str[n - 1];
    for (i = n - 1; i > 0; i--)
        str[i] = str[i - 1];
    str[0] = temp;
}
int main()
{
    char s[50];
    int n, i, z;
    scanf("%d,%s", &n, s);
    z = strlen(s);
    for (i = 1; i <= n; i++)
        move(s, z);
    printf("%s\n", s);
    return 0;
}
```

一样是循环右移。和上面的题对照看。

4-5 以下程序的运行结果是 10,x (1分)

```
#include<stdio.h>
struct n{
  int x;
  char c;
};
void func(struct n b)
{  b.x = 20;
   b.c = 'y';
}
int main()
{
  struct n a = {10, 'x'};

  func(a);
  printf("%d,%c", a.x, a.c);   /* 输出数据之间没有空格分隔 */

  return 0;
}
```

b 是形参！不改变实参的值！

4-6 写出下面程序段的运行结果 2,3 (1分)

```
struct example{
    struct {
      int x;
      int y;
    } in;
    int a;
    int b;
  } e;

  e.a = 1;
  e.b = 2;
  e.in.x = e.a * e.b;
  e.in.y = e.a + e.b;
  printf("%d,%d\n", e.in.x, e.in.y);  /* 输出数据之间没有空格分隔 */
```

4-8 以下程序的输出结果是 efgh (2分)

```
#inlcude<stdio.h>
int main() {
   char *p=”abcdefgh”,*r; long *q;
   q=(long*)p; q++;  r=(char*)q;
   printf(“%s”,r);
}
```

本题略超纲，long 指的是 long int，和 int 完全相同，指的是 32 位 int。

但只要知道了这个，下面的内容并不超纲！
一个 int 是四个字节，一个 char 是一个字节。将 q 地址转为 int*类型后，+1 代表向后移动一个 Int 也就是四个字节的位置，也就是 e 的位置。

%s 输出时，从当前值一直到遇到\0 为止，就是 efgh。

4-12 执行以下程序：

```
#include <stdio.h>
int main( ){
    int score, bonus;
    score = 60;
    bonus = 3;
    switch(bonus){
    case 1:
        score += 10;
    case 2:
        score += 20;
    case 3:
        score += 30;
    default:
        score -= 9;
    }
    printf("%d", score);
    return 0;
}
```

程序运行结果（即：在屏幕上打印的内容）是 81 (3分)。（注意：要严格按照程序打印的格式填写答案，不得随意增加引号、空格等无关字符，否则不得分。例如 `printf("hello World");` 打印的内容就是 `hello World`，而不是 `"hello World"`。）

注意 case 后面有没有 break。switch()在易错题里有详细总结。

4-13 下面程序是将一个正整数分解质因数。例如，输入72，输出 `72=2*2*2*3*3`。请填空。

```
int main()
{   int First;
    int number,i;
    i=2 ;   First = 1;
    scanf("%d", &number);
    printf("%d=", number);
    while (number!=1)
    {    if (number % i == 0)
         {  if (First)
            {    First = 0|First=0  (1分) ;
                 printf("%d",i);
            }
            else
                 printf("*%d",i)   (1分) ;
            number /= i;
         }
         else i++ ;
    }
    return 0;
}
```

这个 First 只是方便输出的，第一个数前面没有*号，后面几个前面有*号。类似题在期中 1 中已经详细介绍过。

4-56 下面程序段的运行结果是 bcdBCD (2分)。

```
char s[20]= "abcd" ;
char *sp = s ;

puts(strcat(sp+1, "ABCD"+1)) ;
```

没什么，不记得 strcat 怎么用就是 0 分。回去看 PPT!

4-57 下列程序段的输出结果是 4599 (2分)

```
int main()
{ int a=3, b=4, c=5, t=99;
  if(b<a&&a<c) t=a; a=c; c=t;
  if(a<c&&b<c) t=b, b=a, a=t;
  printf("%d%d%d\n", a, b, c);
  return 0;
}
```

这个题是**陷阱题!!!!**
if()后面只有 t=a 这一句！ 后面 a=c, c=t 是一定会执行的!
a = c : a==5,b==4,c==5,t==99
c = t : a==5,b==4,c==99,t==99
t=b,b=a,a=t : b==4,a==5,c==99,t==4

期末全是这样的题，你一想当然一定会掉坑里。
期末只有两种题：
1. 你费了半天劲做出来的难题
2. 你十秒钟就做出来自以为对了的陷阱题

下面这两句话记住，**至少能给你期末加两分**：
陷阱题主要靠的就是缩进来蒙骗你，if,for 后面跟几句让你看不出来，偷偷改动冒泡排序的过程让他排序错误，更有甚者在 for 循环后面直接加一个分号，循环运行空语句。大家小心了!

4-58 写出以下程序段的运行结果。

```
/* 程序段A */
int i, j;
int table[3][2];
for(i = 0; i < 3; i++){
    for(j = 0; j < 2; j++){
        scanf("%d", &table[i][j]);
}
}
for(i = 0; i < 3; i++){
    for(j = 0; j < 2; j++){
        printf("%d#", table[i][j]);
    }
}
```

输入 1 2 3 4 5 6

程序段A的输出结果是 1#2#3#4#5#6# (1分)。

程序段A的输出结果是 1#2#3#4#5#6# (1分)。

```
/* 程序段B */
int i, j;
int table[3][2];
for(j = 0; j < 2; j++){
    for(i = 0; i < 3; i++){
        scanf("%d", &table[i][j]);
}
}
for(i = 0; i < 3; i++){
    for(j = 0; j < 2; j++){
        printf("%d#", table[i][j]);
    }
}
```

输入 1 2 3 4 5 6

程序段B的输出结果是 1#4#2#5#3#6# (1分)。

4-59 以下程序的输出结果是 -3#-1#1#3# (3分)

```
#include <stdio.h>
int main()
{
    int s[4][4],i,j;
    for(i=0;i<4;i++)
        for(j=0;j<4;j++)
            *(*(s+i)+j)=i-j;
    for(j=0;j<4;j++)
        for(i=0;i<4;i++)
            if (i+j==3) printf("%d#",*(*(s+j)+i));
      return 0;
}
```

*(*(s+i)+j) == s[i][j]

4-18 程序功能：从键盘读入数据，写到指定的新文本文件中，
遇到字符‘@’结束读入数据
注意：在下面空中填写代码时，不允许出现空格字符

```
#include<stdio.h>
#include<stdlib.h>
int main(void)
{
   FILE *fp;
   char ch,file_write[80];
   scanf("%s",file_write);
   if((fp= fopen(file_write,"w")|fopen(file_write,"w+") (1分))==NULL)
   {    printf("打开文件失败\n");
        exit(0);
   }
   scanf( "%c",&ch (1分));
   while(ch != '@')
   {
     fputc( ch,fp (1分));
     scanf( "%c",&ch (1分));
  }
   fclose(fp) (1分);
  return 0;
}
```

fopen()的几种方式，fgetc(),fputc(),fgets(),fputs(),fclose(),
还有 gets() puts() getchar() putchar() 都要记住。

# 程序填空题

5-1

```
#include<stdio.h>
#include<string.h>
int main()
{
   char s[80];
   static int count[10];
   int i;
   scanf("%s", s);
   for(i=0;i< strlen(s)  (1分);i++)
       if(s[i]>='0'&&s[i]<='9') count[s[i]-'0']++  (1分);
   for(i=0;i<10;i++) printf("%d",count[i]);
   return  0;
}
```

-‘0’可以让字符转为数字

5-2 简化的插入法排序。以下程序段A和B的功能都是：将一个给定的整数$x$插到原本按升序排列的整数序列中，使结果序列仍然按升序排列。

```
/* 程序段 A */
for (i = 0; i < n; i++) {
    if  (x < a[i])  (1分){
        break;
    }
}
for  (j = n-1; j >= i; j--)  (1分){
    a[j+1] = a[j];
}
 a[i] = x;  (1分)
n++;
```

```
/* 程序段 B */
for (i = n-1; i >= 0; i--){
    if(x < a[i]){
         a[i+1] = a[i];  (1分)
    }else{
        break;
    }
}
 a[i+1] = x;  (1分)
n++;
```

插入元素和循环右移一样，是一定要会的操作。

5-3 计算字符串的有效长度。字符串的有效长度就是有效字符的个数，即数组中第1个 '\0' 前面的字符个数。例如，字符串"Happy"的有效长度是5。

```
#include <stdio.h>

int main()
{
    int k, len;
    char str[81];

    k = 0;
    while ((str[k] = getchar()) != '\n') (1分){
        k++;
    }
    str[k] = '\0'; (1分)
    len = k (1分);

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

    return 0;
}
```

本题有缺陷，没有说明 str 字符串要输入。

5-4 删除字符串中的空格（使用1个数组）。

以下程序段的功能是：将字符串*str*中的所有空格都删除。

```
int i, j
char str[81];

i = j = 0;
while (str[i] != '\0') {
    if (str[i] != ' ' (1分){
        str[j] = str[i];
        j++; (1分)
    }
    i++;
}
str[j] = '\0'; (1分)
```

注意删除后，最后还要补上\0 让他成为完整的字符串。

5-5 大小写字母转换（使用1个数组）。

以下程序段的功能是：将字符串*str*中的小写字母全部转换成大写字母，大写字母全部转换成小写字母，其他字符不变。

```
int i;
char str[81];

i = 0;
while (str[i] != '\0') (1分) {
    if (str[i] >= 'a' && str[i] <= 'z'){
        str[i] = str[i]-'a'+'A'; (1分)
    }
    else if (str[i] >= 'A' && str[i] <= 'Z'){
        str[i] = str[i]-'A'+'a'; (1分)
    }
    i++;
}
```

本题在期中 1 中已经详细讲过。

5-7 本题要求输入一个字符串S和两个字符A和B，补足程序中缺失的代码部分，使运行程序时可以将字符串S中的字符A替换为字符B。

```
#include<stdio.h>
int main()
{
    char s[50],a,b,*p; //程序要实现用字符变量b替换字符串s中的字符变量a
    gets(s);
    scanf("%c %c",&a,&b);
    for( p=s; *p!=0; p++ (1分))
        if( *p==a (1分))
            *p=b (1分);
    puts(s);
    return 0;
}
```

用指针的迭代，和 for 循环对比一下！到底有什么区别？

5-10 本程序在数组中同时查找最大元素和最小元素的下标，分别存放在函数main()的max和min变量中。

```
#include <stdio.h>
void find(int *, int, int *, int *);
int main(void)
{
  int max, min, a[]={5,3,7,9,2,0,4,1,6,8};
  find( a,10,&max,&min (1分));
  printf("%d,%d\n", max, min);
  return 0;
}
void find(int *a, int n, int *max, int *min)
{
  int i;
  *max=*min=0;
  for (i = 1; i < n; i++)
  {
    if (a[i] > a [*max]) *max=i (1分) ;
    if (a[i] < a [*min]) *min=i (1分) ;
  }
}
```

注意 find()函数的参数有两个指针！

5-11 下列程序读入时间数值，将其加1秒后输出，时间格式为：hh: mm: ss，即“小时:分钟:秒”，当小时等于24小时，置为0。

```
#include <stdio.h>
struct {
     int hour, minute, second;
} time;
int main(void)
{
      scanf("%d:%d:%d", &time.hour, &time.minute, &time.second (1分) );
      time.second++;
      if( time.second (1分) == 60){
            time.minute++ (1分) ;
          time.second = 0;
          if(time.minute == 60){
              time.hour++;
              time.minute = 0;
              if( time.hour == 24 (0分) )
                 time.hour = 0;
          }
      }
      printf ("%d:%d:%d\n", time.hour, time.minute, time.second );

      return 0;
}
```

# 编程题


```c
#include<stdio.h>

int main()
{
    int a[100];
    a[0] = 1; a[1] = 1;
    int i;
    for (i = 2; i < 20; i++)
        a[i] = a[i-1] + a[i-2];
    for (i = 0; i < 20; i++)
    {
        printf("%10d",a[i]);
        if (i % 5 == 4)
            printf("\n");
    }
}
```

基础中的基础，斐波那契数期末常考。

考点：1. a[0] = 1, a[1] = 1, a[i] = a[i-1] + a[i-2]

2. %10d 输出占位 10 位的整数

3. 5 个一行：判断循环变量对 5 除的余数

返回

## 7-2 找出总分最高的学生 (10分)

给定N个学生的基本信息，包括学号（由5个数字组成的字符串）、姓名（长度小于10的不包含空白字符的非空字符串）和3门课程的成绩（[0,100]区间内的整数），要求输出总分最高学生的姓名、学号和总分。

### 输入格式：

输入在一行中给出正整数N（≤10）。随后N行，每行给出一位学生的信息，格式为“学号 姓名 成绩1 成绩2 成绩3”，中间以空格分隔。

### 输出格式：

在一行中输出总分最高学生的姓名、学号和总分，间隔一个空格。题目保证这样的学生是唯一的。

### 输入样例：

```
5
00001 huanglan 78 83 75
00002 wanghai 76 80 77
00003 shenqiang 87 83 76
10001 zhangfeng 92 88 78
21987 zhangmeng 80 82 75
```

### 输出样例：

```
zhangfeng 10001 258
```

基础，有的同学要是以为结构不考编程，那就大错特错了。

```c
#include<stdio.h>

struct student
{
    char id[100];
    char name[100];
    int score[3];
    int total;//存下来总分
};//这里必须有分号！！

struct student stulist[1000];

 int main()
{
    int n,i;
    scanf("%d",&n);
    int sum,max=0;
    for (i = 0; i < n; i++)
    {
        scanf("%s%s%d%d%d",stulist[i].id,stulist[i].name,&stulist[i].score[0],&stulist[i].score[1],&stulist[i].score[2]);
        //scanf不能忘了地址符号啊！
        stulist[i].total = stulist[i].score[0] + stulist[i].score[1] + stulist[i].score[2];
    }
    for (i = 1; i < n; i++)
    {
        if (stulist[i].total > stulist[max].total)
            max = i;
    }
    printf("%s %s %d",stulist[max].name,stulist[max].id,stulist[max].total);
 }
```

注意，我这里的%s%s%d%d 中间并没有空格或者 getchar()，系统会自动过滤：

%d: 从当前位置开始，找第一个不为空白的数字输入

%s：从当前位置开始，找第一个不为空白的字符输入，遇到回车结束

返回

## 7-3 找鞍点 (10分)

一个矩阵元素的“鞍点”是指该位置上的元素值在该行上最大、在该列上最小。

本题要求编写程序，求一个给定的$n$阶方阵的鞍点。

### 输入格式：

输入第一行给出一个正整数$n$（$1 \le n \le 6$）。随后$n$行，每行给出$n$个整数，其间以空格分隔。

### 输出格式：

输出在一行中按照“行下标 列下标”（下标从0开始）的格式输出鞍点的位置。如果鞍点不存在，则输出“NONE”。题目保证给出的矩阵至多存在一个鞍点。

### 输入样例1：

```
4
1 7 4 1
4 8 3 6
1 6 1 2
0 7 8 9
```

### 输出样例1：

```
2 1
```

考了四五次的题，其实非常简单，但有的同学一看见二维数组就害怕。令人欣慰的是，期末二维数组必考，习惯就好。

```c
#include<stdio.h>

int a[100][100];

int main()
{
    int n,i,j,k1,k2;
    int find = 0; //标记是否找到了
    int flag; //标记当前是否是鞍点
    scanf("%d",&n);
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            scanf("%d",&a[i][j]);
    for (i = 0; i < n; i++)
        for(j = 0; j < n; j++)
        {
            flag = 1;
            //检查行上是否是最大
            for (k1 = 0; k1 < n; k1 ++)
                if (a[i][k1] > a[i][j])
                    flag = 0;
            //检查列上是否是最小
            for (k2 = 0; k2 < n; k2++)
                if (a[k2][j] < a[i][j])
                    flag = 0;
            if (flag == 1)//是鞍点
            {
                find = 1;
                printf("%d %d",i,j);
            }
        }
    if (find == 0)//没找到
    {
        printf("NONE");
    }
    return 0;
}
```

这里有几个要点：

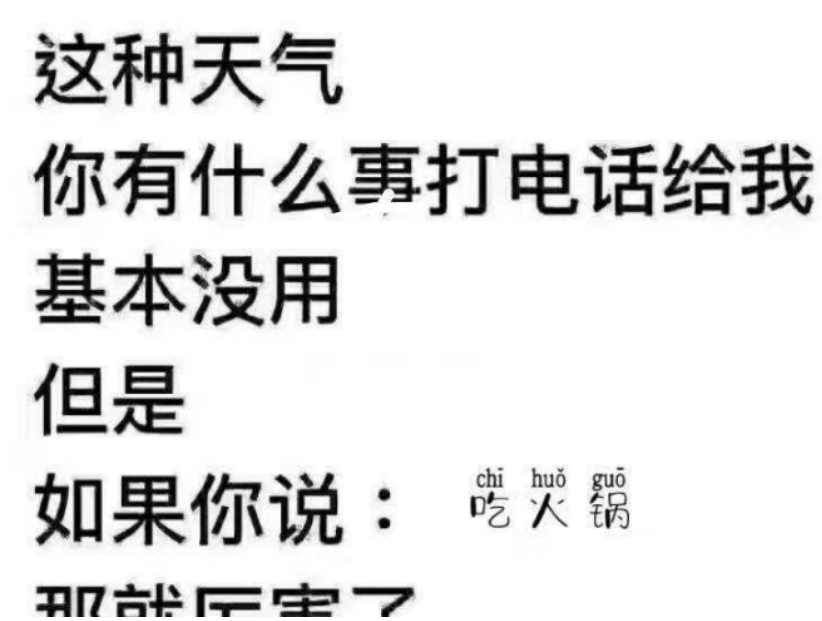
1. 由于最后要输出 NONE，因此需要一个 find 标记判断是否找到过。

2. 对于每个点，循环查看行/列上有没有比其大/小的元素即可。这里我用了一个 flag 标记用于检测。（不要害怕多开变量，但是自己要清楚，自己开的每个变量是干什么的！）

---

返回

7-4 吃火锅 (15分)




以上图片来自微信朋友圈：这种天气你有什么破事打电话给我基本没用。但是如果你说“吃火锅”，那就厉害了，我们的故事就开始了。

本题要求你实现一个程序，自动检查你朋友给你发来的信息里有没有 `chi1 huo3 guo1`。

## 输入格式：

输入每行给出一句不超过 80 个字符的、以回车结尾的朋友信息，信息为非空字符串，仅包括字母、数字、空格、可见的半角标点符号。当读到某一行只有一个英文句点 `.` 时，输入结束，此行不算在朋友信息里。

## 输出格式：

首先在一行中输出朋友信息的总条数。然后对朋友的每一行信息，检查其中是否包含 `chi1 huo3 guo1`，并且统计这样厉害的信息有多少条。在第二行中首先输出第一次出现 `chi1 huo3 guo1` 的信息是第几条（从 1 开始计数），然后输出这类信息的总条数，其间以一个空格分隔。题目保证输出的所有数字不超过 100。

如果朋友从头到尾都没提 `chi1 huo3 guo1` 这个关键词，则在第二行输出一个表情 `-_-#`。

## 输入样例 1：

```
Hello!
are you there?
wantta chi1 huo3 guo1?
that's so li hai le
our story begins from chi1 huo3 guo1 le
.
```

## 输出样例 1：

```
5
```

这个题略难，学有余力的同学可以看一下我的算法，看有没有启发。

## 输入样例 2：

```
Hello!
are you there?
wantta qi huo3 guo1 chi1huo3guo1?
that's so li hai le
our story begins from ci1 huo4 guo2 le
.
```

## 输出样例 2：

```
5
-_-#
```

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

char target[] = "chi1 huo3 guo1";//事先开好一个字符串

int compare(char* a, char* b)
{
    //忘了strcmp函数怎么用？？那就和我一样临时写一个。
    //其实这里应该用的是strstr()函数，直接返回长串中段串首次出现的位置
    //其实最好的方法是在dev上试试strcmp\strstr函数到底怎么用。
    //本函数用于比较一个字符串最前面若干个字符和“吃火锅”是否相等。其中a是字符串，b是吃火锅。
    int n1,n2;
    n1 = strlen(a);
    n2 = strlen(b);
    if (n1 < n2)//长度不够了，不用比了
        return 0;
    int i;
    for (i = 0; i < n2; i++)
        if (a[i] != b[i])//中间必须每一位都相同
            return 0;
    return 1;//相同了
}

int main()
{
    int n = 0;//总句子数量
    int count=0;//吃火锅的数量
    int first = -1;//第一条吃火锅的位置
    char s[100];
    int i,len;
    while (1)//一直循环
    {
        gets(s);//输入一句
        if (strlen(s) == 1 && s[0] == '.')
            break; //跳出循环
        n++;//多了一句
        len = strlen(s);
        for (i = 0; i < len; i++)
            if (compare(s+i,target))//s+i指的是s往后i个字符的字符串！
            {
                if (first == -1)
                    first = n;
                count++;
                break;//注意！一条信息里有就算有两句以上吃火锅，也只算一条！
                //我一开始没想清楚这点，就有一个点答案错误！
            }
    }
    printf("%d\n",n);

    if(count == 0)
    {
        printf("-_-#\n");
    }
    else
    {
        printf("%d %d\n",first,count);
    }

}
```

希望期中 2 题目集最开始的复习顺序推荐公告，还有张老师和我给大家准备的各个文档，能真正帮到大家！预祝大家期末顺利！切记，难题不要慌（难题一般没陷阱），简单题不要乐（多半是陷阱）!