电子科技大学 c 语言复试试题

一 选择题

1.下面程序的输出结果是（　　）

```
int x = 3;
do{
   printf("%d\n",x -= 2);
   }while(!(-- x));
```

解析：x 初始值为 3，第一次循环中运行 printf 函数，参数 x -= 2 的值为 1，输出 1，此时 x = 1，进行判断!(-- x)，x 先自减 1，为 0，取非为 1（真），进行第二次循环，x 先减 2，输出-2，此时 x=-2，判断!(-- x)，x 先自减 1 为-3，取非为 0（假），结束循环。所以输出结果为：

1

-2

2.下面程序的输出结果是（　　）

```
void main(){
      int a[]={1,7,12,15};
      int *p1=a,*p2 = p1++;
      *p1 += *p2++;
      printf("%d %d",*p1,*p2);
}
```

解析：首先定义指针 p1 指向数组 a 首地址，然后定义指针 p2，也指向数组首地址，然后 p1 自加，也就指向了数组第二个元素。*p1 += *p2++;语句先将 p2 指向的第一个元素的值加到 p1 指向的第二个元素的值上，也就是第二个元素值为 8，然后 p2 自加，指向第二个元素（*p2++，*与++优先级相同，从右自左结合，先与++结合，表示语句执行完后 p2 指向下一个元素，然后与*结合，表示 p2 现在所指向的第一个元素的值），所以最后输出结果为：8 8

3.下面程序的输出结果是（　　）

```
int func(int *p){
        return (*p-- = 3) - 1;//(*p-- = 3)可以拆成两步，*p=3; p--;
}

void main(){
        int arr[]={10,7,5};
```

```
        int *p = arr + 1;
        printf(“%d”,func(p) + *p);
}
```

解析：p 指针首先指向数组 arr 的第二个元素，首先调用 func 函数，将实参指针 p 指向的地址传递给函数形参 p，形参执行*p-- = 3，使得 p 指向的数组第二个元素值为 3，然后形参 p 自减指向第一个元素，但是实参 p 不变，还是指向第二个元素，此时 func 返回的值为 3 - 1 = 2，然后 2 + *p，这里实参 p 指向第二个元素，值为 3，所以结果为：5

4.下面程序的输出结果是（　　）

```
void main(){
    int i = 1;
    switch(i){
        printf("hello ");
        case 1:printf("Hi ");
        case 2:printf("Bye ");
    }
}
```

解析：i 的值为 1，所以直接从 case 1 后面的语句开始执行，输出 Hi ，由于这里没有 break;不会跳出 switch 语句，所以继续往下执行，输出 Bye ，所以最后结果为：Hi Bye

5.下面程序的输出结果是（　　）

```
void main(){
    int a,b = 0;
    static int c[10]={9,2,3,4,5,6,7,8,0,1};
    for(a = 0;a < 10;a ++)
        b += c[a];
    printf("%d",b);
}
```

解析：程序遍历数组 c，将 c 的每个元素的值累加到 b 上，所以最后结果为：45

6.运行以下程序后文件的内容为（　　）

```
void main()
{
    char str[100];
    FILE *p1,*p2;
    gets(str);
    p1 = fopen(str,”w”);
    p2 = fopen(str,”w”);
    fputc(‘A’,p1);
```

```
    fputc('B',p2);
    fclose(p1);
    fclose(p2);
}
```

解析：文件指针 p1,p2 分别打开文件，先使用 p1 往文件输出 A，p2 此时指向文件头，所以用 p2 往文件输出 B，覆盖了原本的 A，所以文件中的内容为 B。

7.下面程序的输出结果是（　　）

```
long fib(int n)
{
    if(n > 2) return (fib(n - 1) + fib(n - 2));
    else return 1;
}
void main(){
    printf("%d\n",fib(3));
}
```

解析：调用 fib(3)，由于 3>2，所以返回 fib(2)+fib(1)，fib(2)和 fib(1)都返回 1，所以最后结果为：2

8.下面程序的输出结果是（　　）

```
void main()
{
    char c = 48;
    int i,mark = 01;
    for(i = 0;i < 5;i ++)
    {
        printf("%c",c|mark);
        mark = mark << 1;
    }
}
```

解析：C=48 转换成二进制数就是 110000，mark 初始为八进制 1，循环执行 5 次，每次先输出 c|mark（按位或运算）对应的 ASCII 字符，然后 mark 左移 1 位，即乘以 2。第一次循环 c 为 110000，mark 为 1，c|mark 为 110001，即十进制 49，对应 ASCII 字符‘1’，然后 mark 左移为 10；第二次循环 c|mark 为 110010，即十进制 50，对应 ASCII 字符‘2’，然后 mark 左移为 100，以此类推，最后输出为 12480。

9 程序如下，输出结果为：

```
void main(){
    int x = 9;
    do{
        printf("%d", x);
    }while(!x--);
}
```

输出结果：9

10 //这个题有些地方记不清，50 60 70 是我随便写的，不过不影响做题，考运算符优先级

```
struct Stu{
    Int num;
    Int score;
};
void main(){
    struct Stu stu[3] = {{50,50}, {60,60}, {70,70}};
    struct Stu *p;
    p = stu[0];
}
```

问哪个选项是不对的？

选*p.num= 50，因为.的优先级高于*，而.是直接取值，改为(*p).num

11 程序如下：

```
swap(int *a, int *b)
{
    int t = *a;
    *a = *b;
    *b = *a;
}
```

主函数中定义整型变量 x，y，则下面函数调用方式正确的是（ ）。

A. swap(&x, &y); B. swap(x, y);

C. swap(*x, *y); D. swap(y, x)

答案：A

12 以下程序片段：

```
int i, j, k;
i = 5; j = 10; k = 1;
printf("%d;%d\n", k>i<j, i<j==j<k);
```

输出结果为（ ）。

A. 0;1　　B. 1;1　　C. 1;0　　D. 0;0

答案 C

解析：k>i<j 可以分解为 k>i || i<j，显然为真，则为 1，i<j==j<k，为假，则为 0

13 以下程序片段：

```
int main() {
  char *point[] = {"one", "two", "three", "fourth"};
  point[2] = point[3];
  point[2] += 2;
  printf("%s", point[2]++);

  return 0;
}
```

输出结果为（ ）。

A. ee　B. rth　C. ree　D. urth

答案 D

解析：首先从 point 处开始，先与[]结合，因为其优先级[ ]比*高，所以 point 是一个数组，然后再与*结合，说明数组里的元素是指针类型，然后再与 char 结合，说明指针所指向的内容的类型是字符的，所以 point 是一个由返回字符串型数据的指针所组成的数组。再看题， point[2] = point[3];就是让 point[2]指针指向 point[3]指向的元素即为 fourth。然后 point[2] += 2;也就是指针移动两个地址。指向 fourth 中的 u，最后 printf("%s", point[2]++);先输出 point[2]指向的字符串，urth,再地址+1。

14 下列一维数组定义中，错误的是（ ）。

```
A. int dat[] = {1, 2, 3};
B. int len = 5, dat[len];
C. int dat[5];
D. int dat[5] = {1, 2, 3};
```

答案：B(其实按照新标准 B 没错)

15

```
long fib(int n) {
   if (n > 2)
   {
      return (fib(n - 1) + fib(n - 2));
   } else
   {
      return (2);
   }
}
void main()
{
   printf("%d", fib(3));
}
```

输出结果为（ ）。

A. 2　B. 4　C. 6　D. 8
答案：B

16 以下能正确计算 s=1*2*3*4*5*6*7*8*9*10 的程序段是（ ）。

```
A. do {i = 1; s = 1; s = s * i; i++;} while(i <= 10);
B. do {i = 1; s = 0; s = s * i; i++;} while(i <= 10);
C. i = 1; s = 1; do {s = s * i; i++;} while(i <= 10);
D. i = 1; s = 0; do {s = s * i; i++;} while(i <= 10);
```

答案：C

17 下列程序执行后的输出结果是（ ）。

```
void reverse(int a[], int n) {
 int i, t;
 for (i = 0; i < n; i++)
  {
   t = a[i];
   a[i] = a[n - 1 - i];
   a[n - 1 - i] = t;
  }
}
int main()
 {
 int b[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
 int i, s = 0;
 reverse(b, 8);
 for (i = 6; i < 10; i++)
 {
   s += b[i];
 }
 return 0;
}
```

A. 22 B. 24 C. 26 D. 28
本题题目出错，无论选什么选项均得分
注：本题 **reverse** 函数的原意是反转指定长度的部分数组，但是 **for** 循环终止条件写错，导致本道题出错。
怎么改正确呢！把 reverse 函数 for 循环的 n 改成 n/2，然后主函数，加一个 printf(“%d”,s)答案就为 22 了选 A

18 下列程序执行后的输出结果是（ ）。

```
void func(int *a, int b[]){
 b[0] = *a + 6;
}
void main()
{
```

```
 int a, b[5] = {0};
 a = 0;
 b[0] = 3;
 func(&a, b);
 printf("%d\n", b[0]);
}
```

A. 6　B. 7　C. 8　D. 9

答案：A

19

```
int i, j;
i = 3 / 2 + 7 / 2 == 5;
j = 45 % 11 + (((7 > 8) ? 15 : 14) == 14);
```

后变量 i，j 的值应为（ ）。

A. i = 0 j = 1　B. i = 1 j = 1　C. i = 0 j = 2　D. i = 1 j = 2

答案：C

20 执行下列程序段后，输出的结果是（ ）。

```
int x = 9;
do{
 printf("%d", x--);
}while (!x);
```

A. 8　B. 9　C. 不输出任何内容　D. 陷入死循环

答案：B

21 设 i、j 和 k 都是 int 型变量，且 i = 7，j = 8，k = 9，则表达式(i * j) / k + 6 - 15 % k 的值是（ ）。

A. 9　B. 8　C. 7　D. 6

答案：D

解析：原式 = 56/9+6-6=6

22 把 316 表示为两个加数的和，使两个加数分别能被 13 和 11 整除。

```
void main() {
 int i = 0, j, k;
 do{
 i++;
 k = 316 - 13 * i;
 } while (     );
 j = k / 11;
 printf("316 = 13 * %d + 11 * %d\n", i, j);
```

```
}
```
A. k / 11　　B. k % 11　　C. k / 11 == 0　　D. k % 11 == 0

答案：B

23 如果 char cc[] = "12345", 则 strlen(cc)的值应为（ ）。

A. 7　　B. 5　　C. 6　　D. 0

答案：B

24 若有以下定义和语句：

```
struct student {
 int age;
 int num;
};
struct student stu[3] = { {15, 100}, {16, 90}, {15, 80} };
void main()
{
 struct student *p = stu;
   ...
}
```
则以下不正确的是（ ）。

A. (p + 1) -> num = 200;　　B. p++　　C. *p.num = 200;　　D. p = stu;

答案：C,应该(*p).num = 200;

25 关于 C 程序的叙述，错误的说法是()

A. C 程序总是从主函数开始执行

B. C 程序中定义的第一个函数是主函数

C. 在主函数中可以调用其他函数

D. 一个 C 程序可以包括多个函数

答案：B

26 设 a、b 和 c 都是 int 型变量，且 a=7，b=8，c=9，则表达式（a*b）/c+6-14%c 的值是（）

A.6　　B.7　　C.8　　D.9

答案：B

27 下列一维数组说明中，错误的是（）

A. int a[]={1,2,3};

B. int a[5];

C. int n=5, a[n];

D. int a[5]={1,2,3};

答案：C

28 设有定义 int a,*p;下列赋值语句中，正确的是（）

A. p=a;　　B.p=&a;　　C.p=*a;　　D.*p=&a;

答案：B
29 如下程序片段

```
int a=8, b=3;
float f;
f=a/b;
printf("%.2f\n", f);
```

则该程序片段的输出是（）
A. 2　　B.2.00　　C.2.67　　D.不确定
答案：B
30 程序如下：

```
void main(){
    int i=1;
    switch(i){
        printf("Hello");
        case 1:printf("Cheng ");
        case 2:printf("Du ");
    }
}
```

程序的运行结果是（）
A. Hello Cheng　B.Hello Du　C.Hello Cheng Du　D.Cheng Du
答案：D

二 填空题

1. 选择排序法的链表实现
(答案大概是 while(p->next) ..p=p->next head=head->next return head)

```
//
    p=(struct node*)malloc(sizeof());
    p->next=head;
    head=p;//新建一个头结点
    p=head->next;
    k=q=head;
    while(...)
    {
        //初始化，记不得了，大体结构就这样
            while(p)
            {
                if(q->next->data<p->data)
                  q=p;
                p=p->next;
            }
            If(q!=k)
            {
```

```
                r=q-;
                q->next=r->next;
                p->next=r;
            }
        }
            P=head;
            head=head->next;
            free(p)
            return head;

//
 (答案大概是 while(p->next) ..p=p->next head=head->next return head)
1)每次将最小的元素挪到链表的头部。即 min.next=head;
while(p->next) {
for(p=head;p->next!=NULL;p=p->next){
            if(p->next->data<q->data){
                min=p->next;
                                }}

Min->next=head;
Head=min;
            }
```

2）先建立一个 newHead，循环原链表找到最小的元素，用连接到 newHead 的后面，相当于拆成了两条链。之后找到的最小值都加入新链表的末尾。代码差不多吧。

代码思路：

方法一：换值不断链

```
#include <stdio.h>
#include <stdlib.h>
typedef struct LNode{
    int data;//数据域
    struct LNode *next;//指针域，指向其后继结点
}LinkNode;

void creat_L(LinkNode * &L, int a[], int n)
{
    int i;
    LinkNode *s, *r;
    L = (LinkNode *)malloc(sizeof(LinkNode));
    r = L;
    for(i=0; i<n; i++)
    {
        s = (LinkNode *)malloc(sizeof(LinkNode));
        s->data = a[i];
        r->next = s;
```

```
            r = s;
        }
        r->next = NULL;

}

void print_L(LinkNode * &L)
{
        LinkNode *p = L->next;
        while(p != NULL)
        {
            printf("%d ", p->data);
            p = p->next;
        }
}
//返回类型为结点指针（地址）
LinkNode * MinList(LinkNode *L){
        LinkNode *minp=L,*p=L->next;//minp 指向头结点
        while(p!=NULL){
            if((minp->data)>(p->data)){
                minp=p;
            }
            p=p->next;
        }
        return minp;
}

void SortList(LinkNode *&L){
        LinkNode *j,*r=L->next;//j 指向链表中最小的结点，r 指向链表的当前结点
        int temp;//用于交换位置，设数据为整数
        while(r->next!=NULL){
            j=MinList(r);
            if(j->data!=r->data){
                temp=r->data;
                r->data=j->data;
                j->data=temp;
            }
            r=r->next;//如果当前结点数据就是最小值直接后移一位
        }
}

int main()
{
        int a[] = {3, 8, 3, 9, 8, 5, 0, 3, 4, 10};
```

```
    LinkNode *L;
     creat_L(L, a, 10);
     SortList(L);
     print_L(L);
    return 0;
}
```

方法2：尾插法，断链

```
#include<stdio.h>
#include<stdlib.h>
#define N 8
typedef struct node{
  int data;
  struct node *next;
}ElemSN;
ElemSN *Createlink(int *a)
{
  ElemSN *h=NULL,*np;
  for(int i=N-1;i>=0;i--)
  {//逆向建链
    np=(ElemSN *)malloc(sizeof(ElemSN));
    np->data=a[i];
    np->next=h;
    h=np;
  }
  return h;
}
ElemSN *Sortlink(ElemSN *head)
{
  ElemSN *q,*p,*pm,*qm,*h=NULL,*tail;
  while(head)
  {
    for(q=pm=head,p=head->next;p;q=p,p=p->next){
    if(p->data<pm->data)
    {//找到最小值结点
      pm=p;
      qm=q;

      }
    }
    if(pm!=head)
    {//挂链
      qm->next=pm->next;
    }
    else{//挪头指针
```

```
    head=head->next;
   }
   pm->next=NULL;//要尾插的结点指针域赋空
   if(!h)
   {//新头指针指上来
    h=tail=pm;
   }
   else{//挂链挪动指针
    tail=tail->next=pm;
   }
  }
  return h;
  }
void Printlink(ElemSN *head)
{
  ElemSN *p;
  for(p=head;p;p=p->next){
   printf("%d",p->data);
  }
}
int main()
{
  int i;
  int *a;
  ElemSN *head=NULL;
  a=(int *)malloc(N*sizeof(int));
  printf("Please input a[i]:");
  for(i=0;i<N;i++)
  {
   scanf("%d",a+i);
  }
  //创建链表
  head=Createlink(a);
  //调用函数
  head=Sortlink(head);
  //输出链表
  Printlink(head);
  return 0;
}
```

2 完美乘法，a*b=c,abc 中只出现 0~9 的数字，且每个数字在这个等式中只出现一遍。

(答案大概是 c=a*b while(y>0) y=y/10; if(f[x]!=1) t=1; if(t==0) n++ 这题看着麻烦，其实最简单)

本题出过编程题，放这里意思告诉你，考过填空题，注意下！

原题不清楚了，我尽可能还原代码了，理解思想，题目就不难了

```
#include <stdio.h>
int main(void)
{
    int k,x,count = 0;
    int f[10],s[3];
    int key,y,t;

    for(int i=1; i<99; i++)
    {
     for(int j=100; j<9999; j++)
     {
        key = 0;            //统计数组中不为 1 元素个数
         for(x=0; x<10; x++)     //数组清零测试为下一个数做准备
            f[x]=0;

        k = i*j;
        s[0] = i;
        s[1] = j;
        s[2] = k;
        //统计乘式中 0--9 数字出现的次数并存于数组 f 中
        for(x=0; x<3; x++)
        {
             y = s[x];
             while(y>0)
            {
               t = y%10;
               f[t]++;
               y = y/10;
            }
        }
        //统计数组中不为 1 元素个数
        for(x=0; x<10; x++)
        {
         if(f[x]!=1) key++;
        }

        if(key==0)
        {
         count++;
         printf("%d * %d = %d\n", i, j, k);//打印输出乘式
        }
    }
```

```
    }
     printf("%d", count);
    return 0;
}
```

3 快速排序法求某个数组前 n 个元素第 k 大的数（17 回忆说是快排找第 K 小，其实的确是第 k 小，手动滑稽）

(答案大概是 i<j break low=i+1 high=i-1 ...return a[i])

方法一：从大到小快速排序完以后，直接取第 K 个数

```
#include <stdio.h>
void QuickSort(int arr[],int L,int H)
{
    int ipivot;
    int itemp;
    int icount=1;
    int low = L;                //像我这样把得到枢纽的位置和快速排序放在一起，则 L，H 在这一趟排序中不能变，而改用 low,high 的变化求出枢纽的位置，因为 L，H 还要为下一次循环用。
    int high = H;
    itemp=arr[low];

    if(L < H)                   //一定要有 L < H 判断，不然会有 H 为负值出现
    {
        while(low != high)
        {
            while((low < high)&&(arr[high] > itemp))
                --high;
            if(low < high)
            {
                arr[low] = arr[high];
                ++low;
            }
            while((low < high)&&(arr[low] < itemp))
                ++low;
            if(low <high)
            {
                arr[high] = arr[low];
                --high;
            }
        }

        ipivot = low;     //====ipivot = high,而不是=high+1
         arr[ipivot] = itemp;
```

```
            QuickSort(arr,L,ipivot-1);
            QuickSort(arr,ipivot+1,H);
        }
}

int Max_K(int arr[],int length,int k)
   {
        QuickSort(arr,0,length-1);
        return arr[k-1];
   }
   ////////////////后面的写答案时，不要，舍去，只是为了验证。
int main()
{
      int arr[]={23,1,3,6,5,78,45,7};
      int length;
      int max_k;
      int k;

      length = sizeof(arr)/sizeof(int);
      printf("please input the data of K!\n");
      scanf("%d",&k);
      printf("length:%d\n",length);
      max_k = Max_K(arr,length,k);

     printf("OK,the %dth of arr is %d\n",k,max_k);
     return 0;
}
```

看起来高级一点的方法（网上看的（未看））：

基本思想：

以最后一个元素 x 为轴，把数组分为两部分 Sa 和 Sb。Sa 中的元素大于等于 X，Sb 中元素小于 X。这时有两种情况：

1. Sa 中元素的个数小于 k，则 Sb 中的第 k-|Sa|个元素即为第 k 大数；
2. Sa 中元素的个数大于等于 k，则返回 Sa 中的第 k 大数。

时间复杂度近似为 O(n)

Ps:测试代码你们写一下试试，测试了没毛病的

```
int partition(int *a,int l,int r)
{
      int i=l-1,j=l;
      int x=a[r];
      int temp;

      for(j=l;j<r;j++)
      {
```

```
            if(a[j]>=x) //把比 x 大的数往前放
            {
                exchange(a[j],a[i+1]);
                i++;
            }
        }
        exchange(a[r],a[i+1]);
        return i+1;
}
int k_element(int *a,int l,int r, int k)
{
        if(l>=r)
            return a[l];
        int q=partition(a,l,r);
        if(q==k-1)
            return a[q];
        else if(q>=k)
            return k_element(a,l,q-1,k); //Sa 中元素个数大于等于 k
        else
            return k_element(a,q+1,r,k-(q+1)); //Sa 中元素个数小于 k
}
```

4　100 个人围成圈，从第一个人开始，1~3 报数，数到 3 的人退出，问剩下来的人编号是多少？（后面编程题有源码，放这里意思告诉你，考过填空题，注意下）

(答案大概是 p=a+N ,k=N, p>a+N, i<3 ,i%3, a[i]!=0)

(答案大概是 p=a+N ,k=N, p>a+N, (*p)!=0 ,i%3, a[i]!=0)

5 读程序，写出运行结果或运行意义（10*2）

```
#include <stdio.h>

int main()
{
    char p[10]="abc";
    char q[]="xyz";
    int i,j;
    i=0;
    while(*(p+i)!='\0') i++;//i=3
    j=0;i++;                    //i=4
    while(*(q+j)!='\0')
    {
        *(p+i)=*(q+j);     //p+4  第一次赋值就付给了 p[4],但是 p[3]='\0'哦
        j++;
    }
    printf("%s", p);
    return 0;
```

```
}
```

输出结果应该是 abc

6 今年这道坑人，送分题变成送人头了，和 14 年的正好相反 14 年的那道求 sum=1-x+(x^2/2!)+....+(X^n/n!) 今年结果是 sum=1+x-x^2/2!+x^3/3!-.....+(-1)^(n+1)x^n/n!（读程序）这个题我在编程题总结全了，放这里说一下，考过填空。这里用动态规划思想做！

```
for(i=1; i<=n; i++)
{
    item *= -1*x/i;
    sum += item;
}
```

7.有四道题，每空 3 分，共 72 分，共有四道题，分别是求质因数、删除字符串中的英文和数字并将重复字符减为一个、约瑟夫环、链表逆序。（我后面编程题都总结了，这里大家了解一下，考过填空）

8 下面程序的运行结果是什么

```
void main()
{
    void fun(int a[],int n);
    int a[] = {1,2,4,8};
    int i;
    fun(a,4);
    for(i = 0;i < 4;i ++)
       printf("%d,",a[i]);
}

void fun(int a[],int n)
{
    int i,*p;
    for(i = 0;i < n;i ++)
       p = &a[i];
    *p = 0;
}
```

解析：fun 函数中 for 循环每次将 a[i]的地址赋给指针 p，注意这里没有花括号，循环只执行 p = &a[i];这句，循环结束后，p 指向数组的最后一个元素，然后执行*p = 0;将最后一个元素改为 0，所以运行结果是 1,2,4,0,。

9 下面这段程序的功能是什么？

```
#include<stdio.h>
#include<string.h>
char str[100];
char string[100];

void main()
{
    void fun(int m);
    int m;
    gets(str);
    scanf("%d",&m);
    fun(m);
    printf("%s\n",string);
}

void fun(int m)
{
    int len,i;
    len = strlen(str);
    if(m > len)
    {
        string[0] = '\0';
        return ;
    }
    for(i = 0;str[m - 1] != '\0';i ++,m ++)
        string[i] = str[m - 1];
    string[i] = '\0';
}
```

从 str 字符串第 m 个字符开始截取后面的子串，并复制给 string 字符串，输出 string 字符串。

10 求 2/1+3/2+5/3+8/5+…的前 20 项之和。

```
#include<stdio.h>
void main()
{
    ____   m,k,s = 0,i = 1,j = 2;
    for(k = 1;k < _____;k ++)
    {
        s += _____;
        m =_____; i =_____; j = ____;
    }
    printf("%f\n",s);
}
```

参考答案：

```
#include<stdio.h>
void main()
{
    float   m,k,s = 0,i = 1,j = 2;
    for(k = 1;k <  21   ;k ++)
    {
        s +=  j/i   ;
        m =   i + j   ; i =   j   ; j =   m   ;
    }
    printf("%f\n",s);
}
```

11 一个包含 9 个数的非降序数列存储在数组中，现在插入一个数到合适位置，使序列保持非降序（插入的数大于第一个数，小于第九个数）。

```
#include<stdio.h>
#define N  ____
void main()
{
    int a[N] = {1,2,4,8,16,32,64,128,256};
    int m,i,d;
    scanf("%d",&d);
    for(i = 0;i <  ____  ;i ++)
        if(  _______  ){  ______  ;  ______  ;}
    for(i =  ______  ;i >= m;i --)
        ______  = a[i];
    a[m] = d;
}
```

参考答案：

```
#include<stdio.h>
#define N  10
void main()
{
    int a[N] = {1,2,4,8,16,32,64,128,256};
    int m,i,d;
    scanf("%d",&d);
    for(i = 0;i <  9   ;i ++)     //这里填 N-1 也行
        if(  d < a[i]   ){  m = i   ;  break   ;}
    for(i =  8   ;i >= m;i --)    //这里填 N-2 也行
        a[i + 1]   = a[i];
    a[m] = d;
}
```

12　程序如下：

```
void main()
{
    char p[10]="UESTC";
    char q[]="xyz";  //记不清这是啥了，不影响做题
    int i,j;
    i=0;
    while(*(p+i)!='\0') i++;
    j=0;i++;
    while(*(q+j)!='\0')
    {
        *(p+i)=*(q+j);
        j++;
    }
    printf("%s", p);
}
```

输出为：UESTC

13 该程序功能是什么？输出结果是啥？题目给的输入 n 为 5。

```
void main(){
    char str[][30] = {"turbo c", "turbo br", "dbase", "mdanreiaw", "lfjwae", "jfawlej"};
    int n;

    scanf("%d", &n);
    for (i = 0; i < n-1; i++){
        for (j = i + 1; j <= n-1; j++){
            if (str[i] > str[j])
                swap(str[i], str[j]);
        }
    }
}
```

Ps:字符串不可直接比较，应该用函数，if(strcmp(str[i], str[j]) == 1)，strcmp 比较字符串 str1 和 str2 是否相同。如果相同则返回 0；如果不同，在不同的字符串处如果 str1 的字符大于 str2 的字符，则返回 1，否则返回-1

答：功能：对前 n 个字符串进行字典排序；结果显而易见

14　程序如下：（本题可改编程题）

```
#include<stdio.h>
void main()
{
    int i,j,k,d,flag1,flag2;

    scanf("%d",&d);
    printf("d=%d\n", d);
    for(i=1;i<=100;i++)
    {
        j=i;
```

```
            flag1 = 0;
            while((j>0) && (!flag1))
            {
                k=j%10;
                j=j/10;
                if(k==d)
                    flag1=1;
            }
            if(flag1)
            {
                j=i*i;
                flag2 = 0;
                while((j>0) && (!flag2))
                {
                    k=j%10;
                    j=j/10 ;
                    if(k==d)
                        flag2=1;
                }
                if(flag2)
                    printf("%-5d %-5d\n", i, i*i);
            }

        }
}
```

答：功能为从 1-100 中找出符合以下条件的数，该数和该数的平方中都含有输入的数字 d

15 在 str1 中删除出现在 str2 中的字符，原程序想不起来了，不过是用指针做的。题目的思路就是对于 str1 中的每一个字符，遍历一遍 str2，有则删除。自己编一遍能实现就行~（本题可以当编程题做做）

感谢老习提供的代码，虽然并不是用指针做的，可以当做一个参考

```
#include<stdio.h>
#include<string.h>
main()
{
    char s1[100],s2[100];
    int i,j=0,k=0,len;
    scanf("%s",s1);
    scanf("%s",s2);
    len=strlen(s2);
    for(i=0;s1[i]!='\0';i++)
    {
        while(s1[i]!=s2[j])
            j++;
```

```
            if(j>=len)
            {
                s1[k]=s1[i];//红色标记为填空部分，这个题记不清原代码了，但思想就是我这样。
                k++;
            }
            j=0;
        }
        s1[k]='\0';
        printf("%s",s1);
    }
```

16 链表逆序，这个题 while 那个空给的很诡异，！要你自己写

下面给出两种头插法，我想大多数人应该首先想到用王道上讲的头插法，但是王道的头插法用在这里有个弊端，在程序中已标注。

方法一：王道头插法，从 head->next 开始头插

```
void reverse(List *Head){
    List *P = Head->next;
    List *Q;
Head->next = NULL   //原题缺这一句，造成尾节点 next 不为空，无限输出
    while(P!=NULL){
        Q = P->next;
        P->next = Head->next;
        Head->next = P;
        P = Q;
    }
}
```

方法二：从第三个节点开始头插（第一个节点为 head->next）

```
void reverse(Node *head){
    Node *p = head->next;
    Node *q;
    while(p->next != NULL){
     //提供者说 while 条件里还应该有 P 不能为空，防止链表为空，大家知道
        行了，考试就这么给的空，鬼知道要你填几个条件，随机应变好了~~
        q = p->next;
        p->next = q->next ;
        q->next = head->next;
```

```
            head->next = q;
        }
    }
```

17 如果有两个数，每一个数它的所有约数（除了它本身之外）的和正好等于对方，则称这两个数为互满数，求出 30000 以内所有的互满数，并输出显示。求互满数函数原型（int factor(int j)）（说是填空题，也可改编程题）

```
int factor(int j)
{
    int i, n=0;
    for(i = 1; i<j; i++){
        if(j%i == 0)
            n = n+i;
    }
    return n;
}
int main(void)
{
    int i, j;
    for(i=2; i<10000; i++)
    {
        j = factor(i);
        if(j>i && factor(j)==i)
            printf("%d and %d is HMS\n", i, j);
    }
    return 0;
}
```

18 大概的意思就是输入一个字符串，将连续相同的字符删减为一个，以“*”结尾

（注意这个题，可能会考编程题，后面有个编程，把连续去掉了）

```
void main(){
            char ch_old, ch_new;
            ch_old = '*';
            do{
                scanf("%c", &ch_new);
                if (ch_old == ch_new)
                    continue;
                ch_old = ch_new;
                printf("%c", ch_old);
            }while(ch_new != '*');
    }
```

19

```
int Max(int m,int n){
    int c,x,y;
    x=m;y=n;
    while(m!=0)
    {
        c=n%m;
        n=m;
        m=c;
    }
    return(x*y/n);
}
```

main 函数中的是输入 5 个数字， 然后依次调用 func 函数，
和黄迪明教授的教材中的那个程序其实是一样的。
题目问的是 func 函数的功能：求出 m 和 n 两个数字的最小公倍数
main 函数的功能：求出 5 个数字的最小公倍数

20 程序大致如下：

```
#include <stdio.h>

void func(char *str1,char *str2,int m){
    int len1 = 0,len2 = 0;
    int i = 0,k = 0;
    while (str1[i] != ' '){
        ++i;
        ++len1;
    }

    i = 0;
    while (str2[i] != ' '){
        ++i;
        ++len2;
    }

    for (int j = 0; j<len2; ++j){
        for (i=len1+j; i>=m; --i){
            str1[i+1] = str1[i];
        }
        str1[m] = str2[k];
        ++m;
        ++k;
    }
    str1[len1+len2] = '\0';
}
```

```
int main()
{
    char s1[50] = "helloworld ";
    char s2[20] = "plato ";
    int m;
    scanf("%d",&m);
    func(s1,s2,m);
    printf("%s\n",s1);
    return 0;
}
```

功能是：将 str2 插入到 str1，从第 m+1 个位置开始，str1 原本的字符往后移。

21 读程序，写出功能，并且指出需要改正的地方

```
#include <stdio.h>

int main(){
    float arr[100];//arr可能不够用，需要用动态内存
    int n, i, j, m;
    float k; //错误语句 int k;
    scanf("%d", &n);

    for(i = 0; i < n; ++i){
        scanf("%f", &arr[i]);
    }

    for(i = 1; i < n; ++i){
        for(j = 0; j < n - i; ++j){ //错误语句for(j=0;j<n-i-1;++j)
            if(arr[j] > arr[j + 1]){
                k = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = k;
            }
        }
    }
    m = (n + 1) / 2 - 1;
    printf("the Medians is %f\n",arr[m]);
    return 0;
}
```

该程序的功能是：将数组中的元素排序，并求出其中位数

22 randint 是生成【1，n】中的任意一个随机整数， swap 是用于交换的函数

题目要求你说出每个程序的功能， 时间复杂度， 以及缺点和改进。

func1：

```
for (int i = 0;i < n;++i) {
    a[i] = randint(1, n);

    for (int j = 0;j < i;++j) {
        if (a[j] == a[i]) {
            i--;
        }
    }
}
```

功能：给数组 a 从 1 到 n 随机赋值，并且无重复值

时间复杂度：$O(n^2)$

缺点：时间复杂度过高

改进：使用哈希表替代 for 循环

int b[n];　　int count = b[a[i]]++;　　if(count>1)　i--;

Func2：

```
int tag[100] = {0};

for (int i = 0;i < n;++i) {
    a[i] = randint(1, n);

    for (int j = 0;j < i;++j) {
        if (tag[a[i]] == 1) {
            i--;
        } else {
            tag[a[i]] = 1;
        }
    }
}
```

功能：给数组 a 从 1 到 n 随机赋值，并且无重复值

时间复杂度：$O(n^2)$

缺点：时间复杂度，tag[a[i]]=1

改进：使用哈希表替代 for 循环

int b[n];　　int count = b[a[i]]++;　　if(count>1)　i--;

Func3：

```
1   for(i=0;i<n;i++)
2 {   a[i]=i+1;
3       for(j=0;j<n;j++)
4           swap(a[rand(0,n-1)],a[rand(0,n-1)]);
5   }
```

功能：给数组 a 赋值并随机交换数组 a 任意两个值位置

时间复杂度：$O(n^2)$

缺点：a 数组赋值还没完成就开始随机交换数值，容易使得里面的 0 值交换而无意义

```
for(i=0; i<n; i++)
{
      a[i] = i+1;
 for(j=0; j<i+1; j++)

    swap(a[rand(0,i)],a[rand(0,i)]);
}
```

改进：

```
for(i=0; i<n; i++)
      a[i] = i+1;
   for(i=0; i<n-1; i++)
```

```
{
 int j = rand(i,n-1);
 swap(a[i],a[j]);
}
```

23 有一个 student 结构体数组 Students[]，结构体里面有三个数据 num、age、height，现在要对该数组进行升序排序，若 num 相等，则比较 age，若 age 相等，则比较 height。（具体代码我忘记了，就是一个嵌套的三重循环，需要填空的地方都是填入 if 语句的判断条件）（6 个）

```c
#include<stdio.h>

typedef struct Student {
    int sno;
    int age;
    int height;
}stu;

int main() {
    int n;
    scanf("%d", &n);
    stu s[n], tmp;

    for (int i = 0;i < n;++i) {
        scanf("%d %d %d", &s[i].sno, &s[i].age, &s[i].height);
    }
```

```c
for (int i = 0;i < n - 1;++i) {
    for (int j = 0;j < n - i;++j) {
        if (s[j].sno > s[j + 1].sno) {
            tmp = s[j];
            s[j] = s[j + 1];
            s[j + 1] = tmp;
        } else if (s[j].sno == s[j + 1].sno){
            if (s[j].age > s[j + 1].age) {
                tmp = s[j];
                s[j] = s[j + 1];
                s[j + 1] = tmp;
            } else if (s[j].age == s[j + 1].age) {
                if (s[j].height > s[j + 1].height) {
                    tmp = s[j];
                    s[j] = s[j + 1];
                    s[j + 1] = tmp;
                }
            }
        }
    }
}
```

24 填空题，补充一个函数，该函数的功能是将一个字符串第 n 个位置起长度为 len 的字符串删掉。（伙伴们可以自由书写，具体的程序记不得了 TAT）

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

void Delete_str(char *str, int dlen, int pos){
	int len = 0;
	int i,j;
	char *tmp1 = str;
	char *tmp;
	 len = strlen(str);

	 if(pos<0 || pos>=len){
	    printf("要删除的位置不存在，什么字符都不删除。\n");
	}else{
		for(j = 0; j<dlen; j++){
			tmp = tmp1;
			for(i=0; i<len-1; i++)
			  {
				if(i >= pos)
				{
				 *tmp = *(tmp+1);
				}
				tmp++;
			  }
			*tmp = '\0';

			 len--;
		}

	}
	 printf("%s",str);

}

int main(void){
	int a=0;
	char s1[11] = "helloword";
	printf("%s\n",s1);
	Delete_str(s1, 3, 4);
	return 0;
}
```

25 这道题目其实是 leetcode 上面的题目

92. 反转链表 II

难度 中等 112 收藏 分享 切换为英文

题目描述 评论 (77) 题解 提交记录

反转从位置 *m* 到 *n* 的链表。请使用一趟扫描完成反转。

说明:

1 ≤ *m* ≤ *n* ≤ 链表长度。

示例:

输入: 1->2->3->4->5->NULL, *m* = 2, *n* = 4
输出: 1->4->3->2->5->NULL

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

typedef struct Node{
    int data;
    struct Node* next;
}Node;

Node * reverse(Node *L,int m,int n){
    Node *pre,*p,*q,*t;
    int i=1;

    p=L->next;
    pre=L;
    while(i<m){
        pre=p;
        p=p->next;
        ++i;
    }
    t=p;

    if(m<n){
        p=p->next;
        ++i;
    }

    while(i<=n){
        q=p;
        p=p->next;
        ++i;
        q->next=pre->next;
        pre->next=q;
    }
    t->next=p;
    return L;
}
```

```
38
39  void main(){
40      Node *p,*l,*q;
41      int i;
42      p=(Node *)malloc(sizeof(Node));
43      p->data=0;
44      l=p;
45      q=p;
46      for(i=1;i<=5;i++){
47          p=(Node *)malloc(sizeof(Node));
48          p->data=i;
49          q->next=p;
50          q=q->next;
51      }
52      p->next=NULL;
53      reverse(l,2,4);
54      for(i=1;i<=6;i++){
55          printf("%d_",l->data);
56          l=l->next;
57      }
58  }
```

26　输入一个 n，当 n 是偶数的情况下计算 1/ 2 + 2 / 3 + 3/ 5，奇数的时候计算 1/ 1 + 1/ 3 + 1/ 5…

```
float caleven(int n) {
    int i = 2, j = 1, tmp;
    float res = 0;

    for (int k = 0;k < n;++k) {
        res += (float)j / i;
        tmp = i;
        i = i + j;
        j = tmp;
    }

    return res;
}
```

```
float calodd(int n) {
    float res = 0;
    int tmp = 1;

    for (int k = 0;k < n;++k) {
        res += 1.0 / tmp;
        tmp += 2;
    }

    return res;
}
```

```
int main () {
    int n, r;
    r = scanf("%d", &n);

    if (r != 1) {
        printf("error");
    }else {
        if (n <= 0) {
            printf("error");
        } else {
            if (n % 2 == 0) {
            printf("%f", caleven(n));
            }else {
            printf("%f", calodd(n));
            }
        }
    }

    return 0;
}
```

ps:这几个填空题都有可能考编程题，大家务必注意

27 以下程序执行时从键盘输入”1 2 3 4 5 -1<回车>”，输出结果为：_____(5分)

```
void main(){
    int k=0, n;
    do{
        scanf("%d", &n);
        k=k+n;
    }while(n!=-1);
    printf("k=%d,n=%d\n", k, n);
}
```

参考答案：k=14, n=-1

28 以下程序的输出结果为_____

```
int f(int x, int y)
{
    return (y-x)*y;
}
void main(){
   int a=4, b=5, c=6, d;
   d = f(f(a,b), f(a,c)-f(c,b));
   printf("%d\n", d);
}
```

参考答案：204

29 以下程序的输出结果为_____

```
int arr[3] = {5,3,1};
int *p=&arr[1];
p[1] = *p-1+(p[-1]=3);
```

请写出这段代码执行后 arr 数组各个元素的值

参考答案：3 3 5

30 以下程序的输出结果为_____

```
void main(){
    int a=1, b=2, c=2, t;
    while(a<b<c)
    {
        t=a;
        a=b;
        b=t;
        c--;
    }
     printf("a=%d, b=%d,c=%d\n", a,b,c);
}
```

参考答案：a=1, b=2, c=0

解析：while ((a < b) < c)，1<2 为真，故(a < b)为 1，然后 1<2 成立。交换 a,b 值如此继续，得出结果。

31 输入三个整数，显示出其中最大数，完成下面程序

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  int maxval(int a,int b,int c) {
4      int  ________;
5      if(max<b) max=b;
6      if(max<c) max=c;
7      return ________;
8  }
9  int main() {
10     int a,b,c;
11     printf("Enter there integer numbers:");
12     scanf("%d %d %d",&a,&b,&c);
13     printf("Max=%d\n", ________);
14     return 0;
```

参考答案：

```
#include<stdio.h>
#include<stdlib.h>
int maxval(int a,int b,int c) {
    int max=a;
    if(max<b) max=b;
    if(max<c) max=c;
    return max;
}
int main() {
    int a,b,c;
    printf("Enter there integer numbers:");
    scanf("%d %d %d",&a,&b,&c);
    printf("Max=%d\n",maxval(a,b,c));
    return 0;
}
```

32 下面程序完成两个变量内容的交换，请完成程序。

```
#include<iostream>
using namespace std;
void swap(int *px,int *py) {
    int t;
    ________
    ________
    ________
}
int main() {
    int a=3,b=4;
    cout<<"a="<<a<<" ,b="<<b<<endl;
    swap(_____);
    cout<<"a="<<a<<" ,b="<<b<<endl;
    return _____;
}
```

参考答案：

```
#include<iostream>
using namespace std;
void swap(int *px,int *py) {
    int t;
    t=*px;
    *px=*py;
    *py=t;
}
int main() {
    int a=3,b=4;
    cout<<"a="<<a<<" ,b="<<b<<endl;
    swap(&a,&b);
    cout<<"a="<<a<<" ,b="<<b<<endl;
    return  0(或任意一个整数)    ;
}
```

33 鸭狗共 40 只，脚共有 90 只，以下程序是计算牙狗各多少只，请补全程序

```
void main() {
    int x,y; //代表鸭的只数，y代表狗的只数
    for(x=0; ________;x++){
        y=40-x;
        if(_________)
            printf("duck:%d,dog:%d\n",x,y);
    }
}
```

参考答案:

```
void main() {
    int x,y; //代表鸭的只数，y代表狗的只数
    for(x=0;x<=40;x++){
        y=40-x;
        if(2*x+4*y==90)
            printf("duck:%d,dog:%d\n",x,y);
    }
}
```

34 下面函数计算指针 p 所指的字符串长度（字符串实际个数）

```
1 unsigned int MyStrlen(char _______) {
2     unsigned int len = 0;
3     for(;*p!=_____;p++) {
4         len=_____;
5     }
6     return len;
7 }
```

参考答案：

```
1 unsigned int MyStrlen(char *p) {
2     unsigned int len=0;
3     for(;*p!='\0'或0;p++) {
4         len=len+1;
5     }
6     return len;
7 }
```

三 修改程序题

1、函数 swap 将两个字符串(字符数组作实参，长度不超过 100)的内容进行交换。

```
Void swap(char*pa，char*pb)
{
char*temp;
temp=pa;
pa=pb;
pb=temp;
}
```

改正：

方法 1:

```
void swap(char *pa,char *pb)
{
        char temp[100];//题目说明最大为 100 字节
        strcpy(temp,pa);
        strcpy(pa,pb);
        strcpy(pb,temp);
```

```
}
```

方法 2（我 tm 非要用指针做）：有点麻烦

```
void swap(char *pa,char *pb)
{
   char *temp, i=0;
   temp = (char *)malloc(100*sizeof(char));
   while(*(pa+i)!='\0')
   {
     *(temp+i) = *(pa+i);
     i++;
   }
    *(temp+i) = '\0';
    i=0;
   while(*(pb+i)!='\0')
   {
     *(pa+i) = *(pb+i);
     i++;
   }
    *(pa+i) = '\0';
    i=0;
   while(*(temp+i)!='\0')
   {
       *(pb+i) = *(temp+i);
       i++;
   }
    *(pb+i) = '\0';
}
```

方法 3：

```
void swap(char **p, char **q)
{
      char *temp;
      temp=*p;
      *p=*q;
      *q=temp;
}
```

2、程序片段为：

```
char pa[]="ABCDE";
char *pb="EFG";
pb[1]='A'; //pb 指向的内容是常量，不能修改
pb=pa;
strcpy(pa,"ABCDEFGXYZ"); //pa 指向内存空间不够
```

pb=“D”;

改正：

1. pb指向一个常量字符串，不可修改串中某个元素。pb[1]=’A’尝试修改常量，出错

2. strcpy函数尝试将一个长度超过pa的字符串复制进pa，造成溢出。

char pa[100]=“ABCDE”;

3、程序段如下.(12 分)

```
char a[] = “House”;
char *b = “House”;
//b[2] = ‘r’; //不能对字符常量赋值
a[2] = ‘r’;
b = a;
b[2] = ‘r’;
//a = b;   a 是一个地址常量，不能把地址赋给一个常量
```

4、程序段如下：

p=malloc(sizeof(int)*10);
q=malloc(sizeof(int)*20);
p=q; free(p); free(q);

修改：

一：由于 malloc 函数返回值为 void *，所以还是应该强制转换为（int *）,虽然我在运行时没有错，但一般都要指定指针类型。

二：由于 p=q,两个指针指向同一个地址,free(p)之后，则该内存空间被释放，则 q 此时就没有具体指向一个地址，free(q)就会出错。

改完后如下：

```
int *p,*q;
p=(int*)malloc(sizeof(int)*20);
q=(int*)malloc(sizeof(int)*10);
…
q=p;//修改的话可以删除这句话
…
free(p);
free(q);// 错了，因为 p=q 后，就指向同一块地址了。内存泄露（p 分配了空间没释放）重复释放同一内存会报错，还剩一块内存没有释放，内存泄露。
```

补充知识：

内存泄露：

一般我们常说的内存泄漏是指堆内存的泄漏。堆内存是指程序从堆中分配的，大小任意的（内存块的大小可以在程序运行期决定），使用完后必须显式释放的内存。应用程序一般使用 malloc，calloc，realloc，new 等函数从堆中分配到一块内存，使用完后，程序必须负责相应的调用 free 或 delete 释放该内存块，否则，这块内存就不能被再次使用，我们就说这块内存泄漏了。

内存泄漏是常见的问题。当以前分配的一片内存不再需要使用或无法访问时，但是却并没有释放它，那么对于该进程来说，会因此导致总可用内存的减少，这时就出现了内存泄漏。尽管优秀的编程实践可以确保最少的泄漏，但是根据经验，当使用大量的函数对相同的内存块进行处理时，很可能会出现内存泄漏。尤其是在碰到错误路径的情况下更是如此。

5 下面程序的功能是将字符串 src 逆序输出，请将下面程序的错误改正，缺少的代码补全。

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
void main()
{
char *src = “hello,world”;
char *dest,*d,*p;
int len,i;
len = strlen(src);
dest = (char *)malloc(len);
p = src[len];
d = dest;
while(len-- != 0)
   d ++ = p --;
printf(“%s”,dest);
}
```

①dest = (char *)malloc(len);改为 dest = (char *)malloc(len + 1);因为要多出一个存储字符串结束符'\0'（或者 dest = (char *)malloc(sizeof(char)*(len + 1));这里 sizeof(char)的值为 1，所以写成 len+1 也行）

②p = src[len];改为 p = &src[len - 1];因为 src[len - 1]才是最后一个字符，数组下标从 0 开始，并且要取地址&赋给指针 p

③d ++ = p --;改为*(d ++) = *(p --);这里应将指针 p 指向的字符赋值给指针 p 所指向的内存，所以要加*，*(d ++)，d 先与++结合（这里不加括号也行，*p++，*和++优先级相同，从右往左结合，先与++结合再与*结合，不过习惯性加上括号可以增加代码可读性，也不容易出错），表示语句结束后自加指向下一个字符，再与*结合，表示 d 所指向的字符。*(d ++) = *(p --);语句作用是先把*p 赋值给*d，

然后 p 自减，d 自加，即{*d = *p;d ++;p --;}复合语句的效果
④在 printf(“%s”,dest);语句前要增加一行语句*d = ‘\0’;设置字符串结束标志

改正后的代码：

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
void main()
{
char *src = “hello,world”;
char *dest,*d,*p;
int len,i;
len = strlen(src);
dest = (char *)malloc(len + 1);
p = &src[len - 1];
d = dest;
while(len-- != 0)
   *(d ++) = *(p --);
*d = ‘\0’;
printf(“%s”,dest);
}
```

6 程序如下：

```
Void main(){
   char data[]="There are some mistakes in the program";
   char *point;
   char array[30];
   int i,length;
   length=0;
   while(data[1ength]!='\0')
     length++;
   for(i=0;i<length;i++,point++)
     *point=data[i];
   array=point;
   printf("%s\n",array);
}
```

经过分析，我们可以看出程序的目的是将这个字符串输出，原程序中的主要错误是在指
针运算前没有赋初始地址，于是进行以下修改：

```
#include<stdio.h>
main()
{
   char data[]="there are some mistakes in the program";
   char *point;
   char array[100];//定义一个数组
```

```
    int i,length;
    length=0;
    while(data[length]!='\0')//求字符串的长度
        length++;
    point=array;//给指针赋首地址，主要修改部分
    for(i=0;i<=length;i++,point++)//输出字符串注意这里是<=，\0也被复制进去了
        *point=data[i];
    printf("%s\n",array);
}
```

7 用 Eratoshenes 筛选法求素数，好像程序是个错的，让你修改其中三行，提高效率

采用Eratasthenes 筛选法求 2-200 之间的素数

```
#include <stdio.h>
(2) void main()
(3) {
(4) int prime[201] = {0}; //用于存储 200 以内的数是否已筛去
(5) int d, i;
(6) for (d = 2;d < 200; d++)
(7) if (prime[d] == 0)
(8) for (k = d + 1; k <= 200; k++)
(9) if (k % d == 0) prime[k] = 1;
(10) for (i = 2; i <= 200; i++)
(11) if(prime[i]==0)
(12) printf("%d\t", i);
(13) }
```

改进方法：

请对第 6、7、8、9行的代码进行修改，使得程序执行效率得到提高。

```
for (d=2; d*d < 200; d++)
  if (prime[d] == 0)
  for (k = 2*d; k<=200; k = k+d)
  prime[k]=1;
```

8 程序如下：

```
void main(){
    float d;

    scanf("%f", &d); //学硕这里是%d
    if (d == 3.5)
        putchar("Y");
    else
        putchar("N");
}
```

问：输入 3.5，输出结果是什么？为什么？如何修改？

题目问了如何修改，总感觉它想让你回答 N，但是验证就是 Y，问的怪怪的，不知道出题人怎么想的，不过这只是我的主观臆断……

答：输出 Y（经 VC6 验证过）。因为恰好输入 3.5，数位由 0 补齐，与 3.5 刚好相等，故输出 Y。但是由于浮点数表示都是不精确的，所以不能用“==”判断相等，有几率出现错误结果。修改方法为 if 条件变成“d 减 3.5”的绝对值小于某一精度。如 abs(d-3.5)<0.0001。

9 以下代码是把一个字符串倒序，如”abcd”倒序后变为”dcba”

```
(1) #include <stdio.h>
(2) int main() {
(3) char *src = "HELLO,UESTC";
(4) char *dest = NULL;
(5) int len = strlen(src);
(6) dest = (char *)malloc(len);
(7) char *d = dest;
(8) char *s = &src[len];
(9) while ( len-- != 0 )
(10) *d++ = *s--;
(11) printf("%s\n", dest);
(12) return 0; }
```

错一个，扣 4 分

(1) 第 6行错误，改为：char *dest = (char *)malloc(len + 1); //要为'\0'分配一个字节的空间;

(2) 第 8行错误，改为：char *s = &src[len - 1]; //指向最后一个字符

(3) 第 10行后面补充： *d = '\0'; //尾部要加'\0'

(4) 补充： free(dest); //应当释放空间，以免造成内存泄漏

10 程序如下：

```
(1) void fun() {
(2) char data[20] = "welcome to CHENGDU and UESTC"; 3 (3) char *point; (4) char array[200];
(5) int i, length = 0;
(6) point = array;
(7) while (data[length] != '0')
(8) length++;
(9) for (; i < length; i++) {
(10) *point = *data;
(11) point++;
(12) data++; }
(13) puts(data);
(14) free(data); }
```

每个 3 分

1. 第(2)行：数组长度不够，可修改为：char data[50] = "welcome to CHENGDU and UESTC";
2. 第(7)行：字符串结尾应该是'\0'而不是'0'。
3. 第(5)或者(9)行：i 没有初始化，应该初始化为 0。
4. 第(12)行：数组 data 的地址不能改变。
5. 第(14)行：data 不是 malloc 分配的，不能用free 释放。

6. 第(9)行：for (; i < length; i++)应该改为 for (; i <=length; i++)，因为上面求出来的 length 并非数组长度。

11 经过各位 2019er 们的讨论，此处有两出错，一处涉及到指针的问题，一处是冒泡排序中的循环初始值问题。（以前的真题第一处错常有，第二处错相当于头一次）

用动态内存分配对数据进行排序

```
int *p, n, i, j;//后面还有很多变量， 但是和解题无关
scanf(“%d”, &n);
//由于是要求动态内存分配， 故此处要加个 malloc， 最后也需要 free
for(i=1;i<n;i++)
//有很多种改法， 其中一种就是 i = 0
   for(j=0;j<n-i-1;j++)
       swap(a[j],a[j+1]);//swap 是简写，代表交换两个数组元素值的代码
for (int i = 0;i < n - 1;++i) {
   printf(“%d ”, p[i]);
}
Printf(“%d”, p[n - 1]);
```

四 简答题

1、设 arr 为整型数组，num 和 item 为整型变量，N=数组元素个数-1。需要查找 item 是否在数组中，如果程序片段为

```
for(num=N;arr[num]!=item;num–);
printf(“%d”,num);
```

可能导致的异常结果是什么?为什么?(8 分)

答：可能的异常结果为出现一个随机负数。

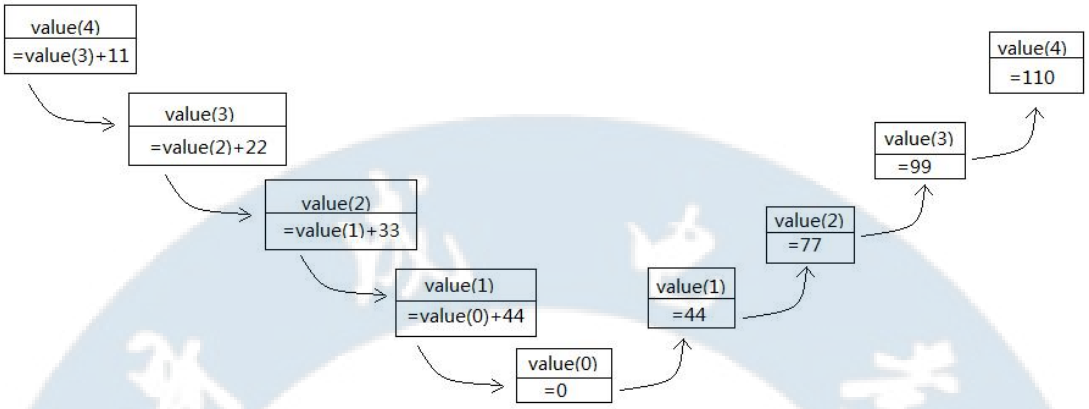
原因：这里有越界的危险哈，有可能在数组中根本就没有给定的值，则 i 会一直在减 1。

结果可能是输出一个负值，因为 item 是个变量，分配了存储空间， arr[-2]也是可以输出的，因为 arr 只是个地址，经过测试会找到 item，printf 他的地址。

2.设有递归函数：

```
int value(int n)
{
  int x;
  if(n==0)
     return 0;
  else
  {
     scanf("%d ",&x);
     return(value(n-1)+x);
  }
```

}
如该函数被调用时，参数 n 值为 4，输入的 x 的值依次为 11，22，33，44，函数调用结束时返回值是多少?并用图描述函数递归执行过程。(10 分)
答：返回值为110



3 数组作为函数参数有三种形式：
1) 实参是数组元素：
2) 形参是指针，实参是数组：
3) 函数的形参和实参都是数组
分别是采用什么参数传递方式?(5 分)
答：1）传递数组元素的值；2）传递数组的起始地址指针；3）传递数组的起始地址

1)实参是数组元素-----传值
2)形参是指针----------传地址(指针)
3)形参和实参都是数组-------传地址

4、采用高度抽象概念有利于程序设计，C 语言中循环语句
do s while（B）;
对应的显示控制结构是什么？请使用伪代码形式（通过条件转移指令）表达。（7 分）
答：（有两个版本答案，第一个是汇编写法）

```
L: s;
cmp B,0
jnz L;
```

或者

```
label: s;
If(B)
goto label;
```

5 从程序执行效率方面考虑，请简述 C 语言采取的一些措施和原因。(15 分)

（ps:这个题大家可以记住大概答案，然后根据详细答案自己扩充，毕竟 15 分在这）

大概答案：

1：指针，可以操作内存

2：位移，可以作位运算

3：API，可以调用系统 API，接近底层

4：宏 define 可以编译的时候替换

详细答案：

1.使用指针：对于指针的理解简单点可以认为类似于汇编中的寻址方式，正是指针的存在使C语言威力无穷。有些程序用其他语言也可以实现，但C能够更有效地实现；有些程序无法用其它语言实现，如直接访问硬件，但C却可以。正因为指针可以拥有类似于汇编的寻址方式，所以可以使程序更高效。

2.使用宏函数：函数和宏函数的区别就在于，宏函数占用了大量的空间，而函数占用了时间。函数调用是要使用系统的栈来保存数据的，如果编译器里有栈检查选项，一般在函数的头会嵌入一些汇编语句对当前栈进行检查；同时，CPU也要在函数调用时保存和恢复当前的现场，进行压栈和弹栈操作，所以，函数调用需要一些CPU时间。而宏函数不存在这个问题。宏函数仅仅作为预先写好的代码嵌入到当前程序，不会产生函数调用，所以仅仅是占用了空间，而使程序可以高效运行。在频繁调用同一个宏函数的时候，该现象尤其突出。

3.使用位操作：位操作可以减少除法和取模的运算。在计算机程序中数据的位是可以操作的最小数据单位，理论上可以用"位运算"来完成所有的运算和操作。一般的位操作是用来控制硬件的，或者做数据变换使用，但是，灵活的位操作可以有效地提高程序运行的效率。

4.循环嵌套中将较长循环设为内存循环，较短循环设为外置循环，以减少cpu跨切循环层的次数，提高程序的运行效率。

6、if(B) s1 else s2; 请问是什么结构，请用显式结构语言表示该程序段，用伪代码写出来(注明条件跳转和强制跳转)。

答：选择结构。

```
Begin
If b! =0        条件跳转
Do S1;
否则 Do S2;
显示结构表示：
if(b) goto L1;
s2; goto L2;
L1: s1;
L2:;
```

7、C 语言中，常量存储在哪儿？static 全局变量和 static 局部变量存储在哪儿？

答：

局部变量：　　栈区
局部静态变量：静态区
全局变量：　　静态区的常量区
全局静态变量：静态区

补充知识：

网上的资料，科普下：

1、栈区（stack）— 由编译器自动分配释放 ，存放函数的参数值，局部变量的值等。其操作方式类似于数据结构中的栈。

2、堆区（heap） — 一般由程序员分配释放， 若程序员不释放，程序结束时可能由 OS 回收 。注意它与数据结构中的堆是两回事，分配方式倒是类似于链表，呵呵。

3、全局区（静态区）（static）—，全局变量和静态变量的存储是放在一块的，初始化的全局变量和静态变量在一块区域， 未初始化的全局变量和未初始化的静态变量在相邻的另一块区域。 - 程序结束后由系统释放

4、文字常量区 —常量字符串就是放在这里的。 程序结束后由系统释放

5、程序代码区—存放函数体的二进制代码。

1、栈，就是那些由编译器在需要的时候分配，在不需要的时候自动清楚的变量的存储区。里面的变量通常是局部变量、函数参数等。

2、堆，就是那些由 new 分配的内存块，他们的释放编译器不去管，由我们的应用程序去控制，一般一个 new 就要对应一个 delete。如果程序员没有释放掉，那么在程序结束后，操作系统会自动回收。

3、自由存储区，就是那些由 malloc 等分配的内存块，他和堆是十分相似的，不过它是用 free 来结束自己的生命的。

4、全局/静态存储区，全局变量和静态变量被分配到同一块内存中，在以前的 c 语言中，全局变量又分为初始化的和未初始化的，在 c++里面没有这个区分了，他们共同占用同一块内存区。

5、常量存储区，这是一块比较特殊的存储区，他们里面存放的是常量，不允许修改。

8 循环语句 for(int i=1;i<n;i++) S；请问是什么结构，请用显式结构语言表示该程序段，用伪代码写出来，汇编语句如何描述，以前是 if，今年考的 for；
答案：循环判断选择结构（这个真不懂，反正就这么回事）
下面答案：

i=1

S_for:

```
if i<n goto S_do
goto S_exit
S_do:
    S
    i=i+1
    goto S_for
S_exit:
```

或者

```
Lable1：if（i<n）goto S;
if（i）=n）goto S1；
S：执行程序；i++；goto Lable；
S1：结束
```

9 根据下面的代码，填写表格。整数算 2 字节，字符 1 字节，指针 4 字节。每个区域的起始地址都是 0，内存按 2 字节编址。

```
int num=2;
void main()
{
char str1[10]={"UESTC"};
char *str2="CHENGDU";
char p;
}
void func(int m)
{
int n=10;
}
```

内存区域	常量或变量名	占用内存大小	内存相对地址
常量区	2	2	0
	10	2	1
	"UESTC"	6	2
	"CHENGDU"	8	5
	10	2	9

全局	num	2	0
main 函数	str1	10	0
	str2	4	5
	p	1	7
func 函数	m	2	0
	n	2	1

这道题难点有二：

① char str1[10]={"UESTC"};　　"UESTC"算字符串常量，但是 str1 对应的内存要占用 10 字节。

② void func(int m)　　形参也是要占用内存的，在函数被调用时申请内存，在函数结束时撤销。

10 从程序效率角度分析下面两段代码。

```
if(B1)                    if(B1)
S1                        S1
if(B2)                    else if(B2)
S2                        S2
…                         …
if(Bn)                    else if(Bn)
Sn                        Sn
```

答：本题要求从程序效率角度考虑，应该默认 B1~Bn 中只有一个事件为真，其它为假。左边程序无论何种情况都要进行 n 次 if 语句判断，而右边语句执行 if 语句判断的次数与 B1~Bn 的情况有关，若 B1、B2、…、B(m-1)为假，Bm 为真，那么只会执行 m 次 if 语句判断，之后的 if 语句不再执行，从而提高了程序效率。

11 若有定义

```
#define SQUARE(x)    ((x)*(x));
int a = 5,b;
```

则执行

```
b = SQUARE(a ++);
```

后，a,b 各为何值？

解析：b = SQUARE(a ++);语句可将宏定义替换掉，即 b = ((a ++)*(a ++));，由于是 a++，先以 a 的原值执行完该语句，a 再自加，也就是{b = a*a;a ++;a ++}复合语句的效果，所以 a = 7,b = 25。

12 请简述 C 语言的隐式类型转换发生的四种情况，并说明每种情况如何转换。

以下是我百度到的内容：

1、算术运算式中，低类型能够转换为高类型。

2、赋值表达式中，右边表达式的值自动隐式转换为左边变量的类型，并赋值给它。

3、函数调用中参数传递时，系统隐式地将实参转换为形参的类型后，赋给形参。

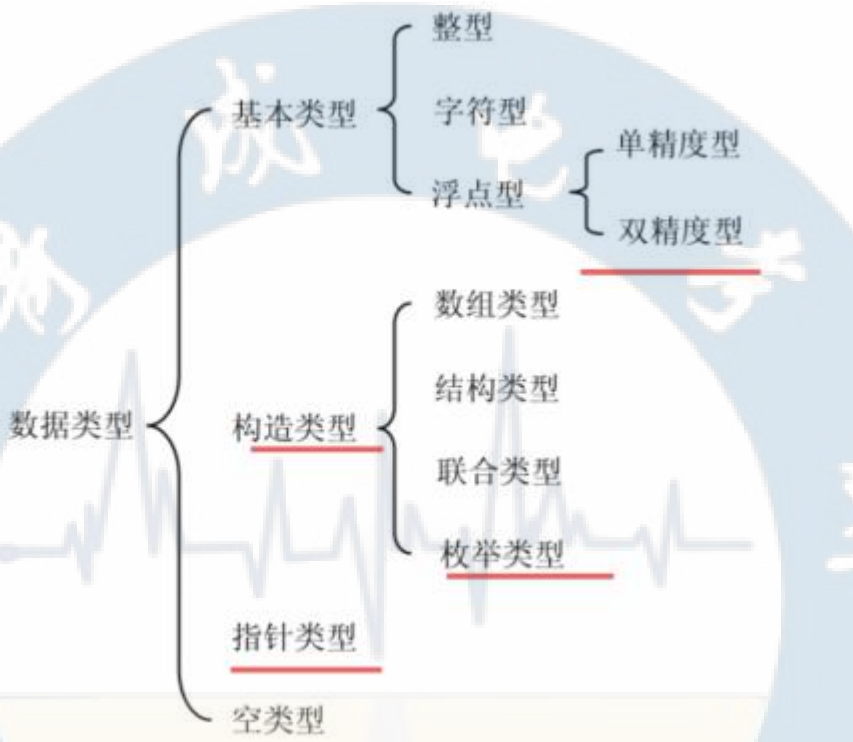
4、函数有返回值时，系统将隐式地将返回表达式类型转换为返回值类型，赋给调用函数。

13 C 语言，除了关键字，还有哪些单词类型（好像是这么问的，百度了一下应该是问的 C 的语法符号）？C 的存储类型关键字有哪四个？

答：（1）标识符，运算符、分隔符、常量、注释符

（2）关键字：auto、extern、register、static

14 四个空 6 分



15 函数中，如何使用与局部变量同名的全局变量？（3 分）

答：使用::（作用域区分符）（3 分）

16 如何使用1 个表达式将 float 类型变量 f 的值四舍五入转换为 long 类型的值？（3 分）

答：(long)(f + 0.5) （3 分）

17 C 程序运行时，不对数组进行越界检查，可能导致什么问题？（5 分）

答：可能导致运行出错（访问不可访问的存储单元）（3 分）或访问和修改其他非数组元素的数据。（2 分）

18 程序片段为

```
int n = 2018;
void main() {
char * p=”COMPUTER”, *q;
```

```
int mm, arr[2018];
char ch2;
  ...
fun(mm);
...
  }
void fun(int nn) {
int mm = 10;
static int snum;
  ...
}
```

当程序执行进入fun 函数时，请列出各个数据（包括常量、变量）在内存中对应存储区的名称和数据的存储顺序以及所占用的存储空间的字节数。假设整数占 2 个字节，字符占 2 个字节，指针占 4 个字节；而内存按 2 个字节进行编址。
存储区名称 1 分，其他每 3 个 1 分

	常量或变量名	占用内存大小
常量存储区	2018	2
	COMPUTER	18
	10	2
全局数据区	n	2
static 数据区	snum	2
main 函数数据区（或活动记录）	p	4
	q	4
	mm	2
	arr	4036
	ch2	2
fun 函数数据区（或活动记录）	nn	2
	mm	2

19 你是产品经理，如果你的产品马上要交付，你会怎么安排测试才能交付给用户使用。（适当写写，开放题型）

答：产品测试一般都是围绕需求为主的产品需求设计说明书 PRD 文档来展开测试的，针对每个功能 点编写测试用例，去验证功能的正确性和完整性。这种方式在正常的开发上线进度下都不会有问题 ，相反是一种很好的验证功能需求实现的方式。但在敏捷开发模式或者因为赶进度的原因，造成产 品测试时间非常紧的情况下，用这种方式就会有点捉襟见肘。

以用户为中心的产品设计已经逐渐的深入人心，产品主题功能要能满足用户的某种诉求或者解 决用户的某个痛点，也有说成是以用户痛点为中心的产品设

计。在这种大背景下，产品开发完成后 ，如果测试时间非常紧，在不能保证产品没有问题但却必须要按时上线的时候，就必须保证产品在 用户使用的场景下没有问题，也就是说优先保证用户使用产品的整个流程当中不会出现问题，这就是基于用户使用场景的产品测试法。 在实际的测试过程当中，最常见的还是基于产品功能的测试，那基于用户使用场景的产品测试 两者之间有什么区别呢？区别一是后者的测试范围更小，忽略了一部分产品后台功能的测试或隐性的功能测试，即只是测试了表面操作性的过程，没有测试底层的功能；区别二是后者的测试是把产 品功能转化成实际用户使用场景下来测试，这就要求测试人员要从普通用户的操作角度出发，而不能受开发人员的影响，以一个初次使用产品的用户角度，来验证产品的功能是否可以在使用过程中提供正常的服务。

20 用 c 语言做网络编程，你将使用 http 还是 TCP/IP 协议来进行网络连接，为什么？

答："http 协议和 TCP 协议的区别是:http 协议是应用层的协议,TCP 协议是传输层的协议,http 协议是建立是 TCP 协议之上的,http 是无状态的的短链接,而 TCP 是有状态的长链接。"像 java 这类语言比较适合应用层 http，c 语言更适合传输层的 tcp/ip。（适当写写）

21 static 全局变量和普通的全局变量有什么区别？

答：static 全局变量初始化一次，防止在其他文件单元中被引用（1 分）；static 变量存放在 static 数据区（2 分）

22 读程序，说明原因

本题有错，应该是缺少了某个语句，如在考试中遇到，说动模拟，该多少是多少，错题都加分

```
#include <stdio.h>
void main(){
   int arr[10], m;
   for(m=0; m<=10; m++)
   {
     arr[m] = m-10;
     printf("%d", arr[m]);
   }
}
```

参考答案：

i -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 只给 1 分

ii -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 -10 只给 3 分

Iii 死循环， arr[10]就是 m，循环每次最后将 m 的值进行了修改（3 分）

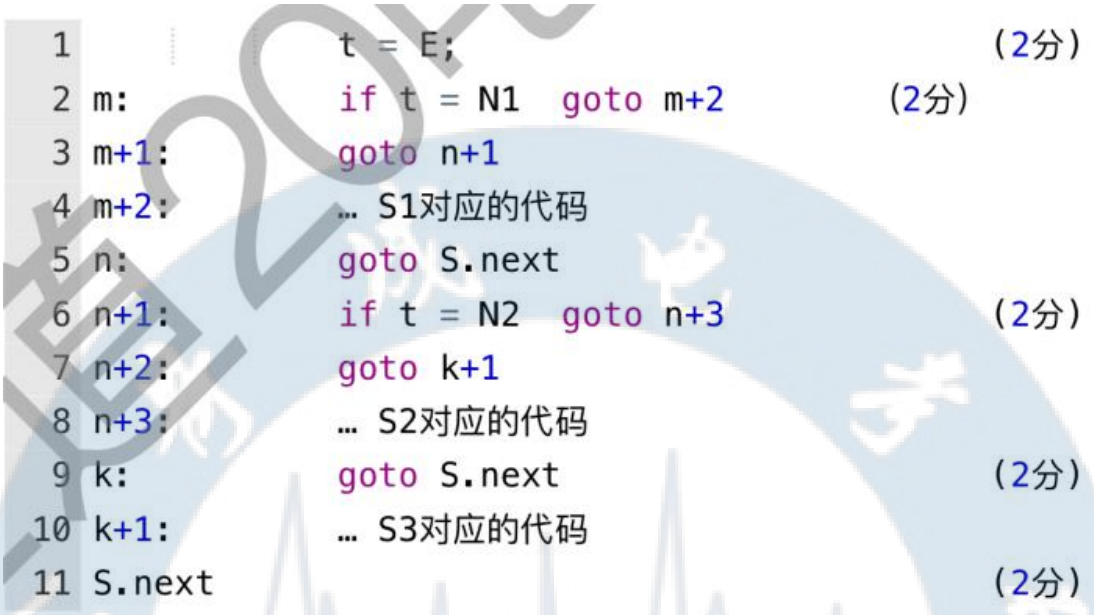
运行结果为：-10 -9 -8 -7 -6 -5 -4 -3 -2 -1 -10-9 -8 -7 -6 -5 -4 -3 -2 -1 -10.....（3 分）

23 采用高度抽象概念有利于程序设计。c 语言中语句

```
switch(E)
{
     case N1:S1; break;
```

```
    case N2:S2; break;
    default S3;
}
```

其中，E 代表常量表达式，N 代表常量，S 代表执行语句。对应的显示控制结构是什么，请用伪代码形式表达。



五、编程题

1. 有一个函数 $y=\begin{cases}-1 & (x<0 \text{或} x>100)\\ 0 & (x=0)\\ 1 & (0<x\le 100)\end{cases}$，写一段程序，输入 x 的值，输出 y 的值。

```
#include <stdio.h>
int main(void)
{
    int x,y;
    printf("请输入一个数：");
    scanf("%d", &x);
    if(x==0)
    {
      y = 0;
    }
    else if(x>0 && x<=100)
    {
      y = 1;
    }
    else    y = -1;
```

```
    printf("y = %d", y);
    return 0;
}
```

2. 输入 3 个数 a, b, c, 按大由到小的顺序输出。

方法一

```
#include <stdio.h>
int main(void)
{
    int a, b, c, min;
    printf("请输入三个数： ");
    scanf("%d %d %d", &a, &b, &c);
    if(a<b)
    {
     min = a;
     a = b;
    }
    else
    {
       min = b;
    }

    if(c>min)
    {
     if(c<a)
     {
        b = c;
        c= min;
     }else
     {
        b = a;
        a = c;
        c = min;
     }

    }
    else
    {
     b = min;
    }
    printf("%d %d %d", a, b, c);
    return 0;
}
```

方法二

```
#include <stdio.h>
int main(void)
{
      float a,b,c,t;
     printf("please enter a,b,c:");
      scanf("%f,%f,%f",&a,&b,&c);
      if(a<b)
       {
       t = a;
       a = b;
       b = t;
       }
      if(a<c)
        {
       t = a;
       a = c;
       c = t;
       }

     if(b<c)
        {
       t = b;
       b = c;
       c = t;
       }

    printf("%7.2f%7.2f%7.2f\n",a,b,c);
    return 0;
}
```

3. 输入 4 个数，输出 4 个数中的最大值、最小值。

方法一

```
#include <stdio.h>
int main(void)
{
    float a, b, c, d, max, min;
```

```
    printf("请输入四个数 a b c d:");
    scanf("%f %f %f %f", &a, &b, &c, &d);
    if(a>b)
    {
     max = a;
     min = b;
    }
    else
    {
     max = b;
     min = a;
    }

    if(c>d)
    {
      if(c>max)
         max = c;
      if(d<min)
         min = d;
    }
    else
    {
     if(d>max)
        max = d;
     if(c<min)
        min = c;
    }

    printf("max=%5.2f min=%5.2f", max, min);

    return 0;
}
```

方法二

```
#include <stdio.h>

int main(void)
{
    float    t,a,b,c,d,max,min;
     printf("请输入四个数:");
     scanf("%f,%f,%f,%f",&a,&b,&c,&d);
       if (a<b)
         { t=a;a=b;b=t;}
       if (a<c)
```

```
        { t=a;a=c;c=t;}
      if (a<d)
        { t=a;a=d;d=t;}
      if (b<c)
        { t=b;b=c;c=t;}
      if (b<d)
        { t=b;b=d;d=t;}
      if (c<d)
        { t=c;c=d;d=t;}
      printf("最大值和最小值分别为: \n");
       printf("max=%5.2f     min=%5.2f",a,d);

    return 0;
}
```

4. 输入成绩，要求输出成绩等级 A、B、C、D、E 或 error。90 分以上为‘A’，80~89 分为‘B’，70~79 分为‘C’，60~69 分为‘D’，0~60 分为‘E’。若输入成绩低于 0 分和高于 100 分则输出 error。

方法一

```
#include <stdio.h>

int main(void)
{
    float score;
    printf("请输入成绩： ");
    scanf("%f", &score);

    if(score>100 || score<0)
    printf("error");
    else if(score>=90) printf("成绩等级为 A");
    else if(score>=80) printf("成绩等级为 B");
    else if(score>=70) printf("成绩等级为 C");
    else if(score>=60) printf("成绩等级为 D");
    else printf("成绩等级为 E");

    return 0;
}
```

方法二

```
#include <stdio.h>

int main(void)
{
```

```
    float score;
    char grade;
    printf("请输入学生成绩:");
    scanf("%f",&score);
    if (score>100||score<0)
   printf("error\n");
   else
    {
     switch((int)(score/10))
     {
        case 10:
         case 9: grade='A';break;
         case 8: grade='B';break;
         case 7: grade='C';break;
         case 6: grade='D';break;
         case 5:case 4:case 3:case 2:case 1:
         case 0: grade='E';
      }
       printf("成绩是  %5.1f,相应的等级是%c.\n ",score,grade);
    }
   return 0;
}
```

5.有一个函数：

$$y=\begin{cases} x & (x<1) \\ 3x-11 & (x=10) \\ 2x-1 & (1\le x<10\text{或}x>10) \end{cases}$$

写一段程序，输入x的值，输出 x，y的值。

```
#include <stdio.h>
int main(void)
{
   int x,y;
   printf("输入 x:");
   scanf("%d",&x);
   if(x<1)                    /* x<1 */
   { y=x;
     printf("x=%3d,     y=x=%d\n" ,x,y);
   }
   else    if(x<10||x>10)            /* 1=<x<10 */
     {
      y=2*x-1;
      printf("x=%d,    y=2*x-1=%d\n",x,y);
     }
     else                  /* x>=10    */
     {
```

```
        y=3*x-11;
        printf("x=%d,   y=3*x-11=%d\n",x,y);
      }
     return 0;
}
```

6. 输出 300~400 之间的全部素数，并按每行 5 个数输出

方法一

```
#include <stdio.h>
int main(void)
{
   int i, sum = 0, j;

   for(i=300; i<=400; i++)
   {
       for(j=2; j<i; j++)
          if(i%j == 0)
            break;
       if(j>=i)
       {
          sum++;
          printf("%d ", i);
          if(sum%5==0)
          printf("\n");
       }

   }
    return 0;
}
```

方法二

```
#include<stdio.h>
int panduan(int n);
int main()
{
  int n, flag, sum=0;
  for(n=300;n<=400;n++)
  {
    flag=panduan(n);
    if(flag == 1)
    {
       printf("%d ",n);
       sum++;
       if(sum%5==0)
```

```
        printf("\n");
    }
  }
  return 0;
}

int panduan(int n)
{
  int i,temp;
  temp=1;
  for(i=2;i*i<n&&temp==1;i++)
        if(n%i==0)
              temp=0;
       return temp;
}
```

7. 求 Fibonacci 数列的前 20 个数，并将其分 5 行输出

方法一

```
#include <stdio.h>
int main(void)
{
    int f[20], i;
    f[0] = 1;
    f[1] = 1;
    for(i=2; i<20; i++)
    {
      f[i] = f[i-1] + f[i-2];
    }
    for(i=0; i<20; i++)
    {
       printf("%d ",f[i]);
       if((i+1)%5==0) printf("\n");
    }
     return 0;
}
```

方法二

```
#include <stdio.h>
int Fibon1(int n)
{
   if (n == 1 || n == 2)
     {
        return 1;
     }
   else
     {
        return Fibon1(n - 1) + Fibon1(n - 2);
      }
}
int main()
{
   int n = 0;
   int ret = 0;

   for(int i=1; i<=20; i++)
   {
     ret = Fibon1(i);
     printf("%d ", ret);
     if(i%5==0)
     {
          printf("\n");
     }
   }
   return 0;
}
```

8. 求 $\sum_{n=1}^{10}(n!+3)$ 。

方法一

```
#include <stdio.h>
int factorial(int n)
{
   int fl=1;
   while(n!=0)
  {
     fl *= n;
     n--;
  }

  return fl;
}
int main()
{
   int i, sum=0;
   for(i=1; i<=10; i++)
   {
     sum +=    (factorial(i) + 3);
   }
   printf("1!+3+2!+3...+10!+3=%d", sum);

   return 0;
}
```

方法二

```
#include <stdio.h>
int main()
 {
   int s=0,t=1;
   int n;
   for (n=1;n<=10;n++)
   {
     t=t*n;
     s=s+t+3;
   }
   printf("1!+3+2!+3...+10!+3=%d\n",s);

   return 0;
}
```

9. 有一分数序列$\frac{1}{2},\frac{2}{3},\frac{3}{5},\frac{5}{8},\frac{8}{13},\frac{13}{21},\cdots$，求其前 20 项的和。

```
#include <stdio.h>
int main(void)
{
   double i=1, k=2, temp;
   double sum = 0;
   for(int j=0; j<20; j++)
   {
     sum += i/k;
     temp = i;
     i = k;
     k = k+temp;
   }
   printf("sum = %.10f", sum);
   return 0;
}
```

10. 输出小于 500 的所有的“水仙花数”，所谓“水仙花数”是指一个 3 位数，其各位数字立方和等于该数本身。例如，153 是一个水仙花数，因为$153=1^3+5^3+3^3$。

```
#include <stdio.h>
int main(void)
{
   int a, b, c;//个位十位百位数字
   for(int i=100; i<500; i++)
   {
     a = i%10;//个位
     c = i/100;//百位
     b = (i-100*c)/10;
     if(i == a*a*a + b*b*b + c*c*c)
     {
        printf("%d ", i);
      }
   }
   return 0;
```

```
}
```

11. 给一个不多于 6 位的正整数，要求：①求出它是几位数；②分别输出每一位数字；③按逆序输出各位数字，例如原数为 158，应输出 851。

方法一

```
#include <stdio.h>
void divide_c(int n);
int main(void)
{
   int a;
   printf("请输入一个不多于 6 位的正整数：");
   scanf("%d", &a);

   if(a/10==0)//0-9 位数
   {
      printf("该数的位数：1\n");
      printf("这个数为：%d\n", a);
      printf("逆序为：%d", a);
   }else if(a/100==0){      //10-99
      printf("该数的位数：2\n");
      printf("这个数为：%d\n", a);
      printf("逆序为：");
      divide_c(a);
   }else if(a/1000==0){
      printf("该数的位数：3\n");
      printf("这个数为：%d\n", a);
      printf("逆序为：");
      divide_c(a);
   }else if(a/10000==0){
      printf("该数的位数：4\n");
      printf("这个数为：%d\n", a);
      printf("逆序为：");
      divide_c(a);
   }else if(a/100000==0){
      printf("该数的位数：5\n");
      printf("这个数为：%d\n", a);
      printf("逆序为：");
      divide_c(a);
   } else if(a/1000000==0){
      printf("该数的位数：6\n");
      printf("这个数为：%d\n", a);
      printf("逆序为：");
      divide_c(a);
   }else printf("输入不满足条件，请重新输入！\n");
```

```
    return 0;

}

void divide_c(int n)
{
      int t;
      while(n!=0)
      {
            t = n%10;
            printf("%d", t);
           n = n/10;
      }
}
```

方法二

```
#include <stdio.h>
int    main()
{
      int n,m;
      printf("输入一个不多于六位的数字:");
      scanf("%d",&n);
      if(n<1000000&&n>99999)
          printf("它是六位数\n");
       if(n<100000&&n>9999)
          printf("它是五位数\n");
       if(n<10000&&n>999)
          printf("它是四位数\n");
       if(n<1000&&n>99)
          printf("它是三位数\n");
       if(n<100&&n>9)
         printf("它是二位数\n");
       if(n<10&&n>=0)
          printf("它是一位数\n");
          printf("各位数字按逆序输出为:\n");
     while(n!=0)
     {m=n%10;
      n=n/10;
      printf("%3d",m);
     }

     return 0;
}
```

12. 求 5+55+555+5555+55555 的值。（拓展输入一个数 a，求 a+ aa + aaa + aaaa + aaaaa）


```
#include <stdio.h>
//5+55+555+5555+55555
int   main()
{
    int a, tn=0, sn=0;
    printf("请输入一个 0-9 的数：");
    scanf("%d", &a);

    for(int i; i<a; i++)
    {
      tn = tn + a;
      sn = sn + tn;
      a = a*10;
    }

    printf("a+aa+aaa+aaa...= %d", sn);
    return 0;
}
```

13. 输入 10 个整型整数，用起泡法对这 10 个数排序，并该由小到大顺序在屏幕上输出。

```
#include <stdio.h>
int   main()
{
    int i, j, temp, a[10];
    printf("请输入十个整数：\n");

    for(i=0; i<10; i++)
    {
     scanf("%d", &a[i]);
    }

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

```
        for(j=9; j>=0; j--)
        {
          if(a[j]<a[j-1])
          {
             temp = a[j];
             a[j] = a[j-1];
             a[j-1] = temp;
          }
        }
    }

    printf("冒泡排序结果为：");
    for(i=0; i<10; i++)
    {
     printf("%d ", a[i]);
    }

    return 0;
}
```

14. 将一个二维数组 a 的行和列的元素互换（即行列互换），存到另一个二维数组 b 中。

```
#include <stdio.h>
#define M 3
#define N 4

int   main()
{
    int i, j;
    int a[M][N] , b[N][M];
```

```
    printf("请输入若干整数：\n");
    for(i=0; i<M; i++)
        for(j=0; j<N; j++)
            scanf("%d", &a[i][j]);

    for(i=0; i<M; i++)
        for(j=0; j<N; j++)
            b[j][i] = a[i][j];

    printf("二维数组 a 为： ");

    for(i=0; i<M; i++)
    {
        printf("\n");
        for(j=0; j<N; j++)
            printf("%d ", a[i][j]);
    }

    printf("\n");
    printf("二维数组 b 为： ");
    for(i=0; i<N; i++)
    {
        printf("\n");
        for(j=0; j<M; j++)
            printf("%d ", b[i][j]);
    }

    return 0;
}
```

15. 在一个 3*4 的二维数组 a 中，要求编程求出其中值最大的那个元素的值，并输出其所在的行号和列号。

```
#include <stdio.h>
#define M 3
#define N 4
int    main()
{
    int i, j, row, col, max;
    int a[M][N];
     printf("请输入若干整数：\n");
    for(i=0; i<M; i++)
        for(j=0; j<N; j++)
              scanf("%d", &a[i][j]);
    max = a[0][0];

    for(i=0; i<M; i++)
    {
       for(j=0; j<N; j++)
        {
          if(a[i][j]>max)
         {
            max = a[i][j];
            row = i+1;
            col = j+1;
         }
       }
    }
```

```
    printf("数组最大元素为：%d\n", max);
    printf("行：%d  列：%d", row, col);

    return 0;
}
```

16. 输入一行字符，统计其中有多少个单词，单词之间用空格分隔开。

```
#include <stdio.h>
int   main()
{
    int i=0, count=0;
    char str[50];
    printf("请输入一串字符：");
    //scanf("%s", str);注意 scanf 与 gets 区别
    gets(str);
    while(str[i] != '\0')
    {
        i++;
        if(str[i]== '\0' || str[i]==' ')
            count++;
    }
    printf("单词个数为：%d", count);

    return 0;
}
```

17. 有 3 个字符串，要求找出其中最大者。

```
#include <stdio.h>
#include <string.h>
int    main()
{
    char str1[80], str2[80], str3[80], longest[80];
    printf("请输入三个字符串：\n");
    gets(str1);
    gets(str2);
    gets(str3);

    strcpy(longest, str1);
    if(strcmp(longest,str2)<0)
        strcpy(longest, str2);

    if(strcmp(longest,str3)<0)
        strcpy(longest, str3);
     printf("最大字符串是：%s", longest);
    return 0;
}
```

18. 输入两个整数，要求用一个函数求出其中的大者，并在主函数中输出此值。

```
#include <stdio.h>

int Max(int n, int m);
int    main()
{
    int a, b;
```

```
    printf("请输入两个整数：");
    scanf("%d %d", &a, &b);

    printf("两者最大是：%d", Max(a, b));
    return 0;
}

int Max(int n, int m)
{
    if(m>n)
    return m;
    else
    return n;
}
```

关于 Max 还可以

```
int Max(int a,int b)
{
    int c;
    c=a>b?a:b;
    return(c);
}
```

19. 输入两个整数，要求用一个函数求出其最大公约数和最小公倍数，并在主函数中调用该子函数。

方法一：辗转相除法（欧几里德法）

```
#include <stdio.h>
void Common(int n, int m);
int   main()
{
    int a, b;
    printf("请输入两个整数：");
    scanf("%d %d", &a, &b);
```

```
    Common(a, b);
    return 0;
}

void Common(int n, int m)
{
    int temp, a, b, com;
    a = n;
    b = m;
    if(m<n)
    {
     temp = m;
     m = n;
     n = temp;
    }
    //辗转相除法
    while(n!=0)
    {
      temp = m%n;
      m = n;
      n = temp;
    }
    //最大公约数 m

    com = a*b/m;
    printf("最大公约数为：%d  最小公倍数为：%d", m, com);

}
```

方法 2 递归法（本质还是辗转相除法）

```
#include <stdio.h>
int gcd(int a,int b);
void Common(int n, int m);

int    main()
{
    int a, b;
    printf("请输入两个整数：");
    scanf("%d %d", &a, &b);
    printf("最大公约数为：%d", gcd(a, b));
    Common(a, b);
    return 0;
}
```

```
int gcd(int a,int b)                    //自定义函数求最大公约数
{
     if(a%b==0)                          //终止条件
          return b;                      //b 及为最大公约数
     else
          return gcd(b,a%b);
}

void    Common(int n, int m)
{
    int g = gcd(n, m);

    printf("最小公倍数为：%d", m*n/g);
}
```

方法三（穷举法）

```
#include <stdio.h>
void Common(int n, int m);

int    main()
{
    int a, b;
    printf("请输入两个整数：");
    scanf("%d %d", &a, &b);

    Common(a, b);
    return 0;
}


void    Common(int n, int m)
{
    int temp;
    temp = (m>n)?n:m;
    while(temp!=0)
    {
        if(m%temp==0 && n%temp==0)
          break;
       temp--;
    } //temp 就为最大公约数

    printf("最大公约数为：%d  最小公倍数为：%d",temp, m*n/temp);
}
```

方法四：更相减损术

```
#include <stdio.h>
#include <math.h>
void Common(int m, int n);
int main()
{
    int a, b;
    printf("请输入两个整数： ");
    scanf("%d %d", &a, &b);

    Common(a, b);

    return 0;
}
void Common(int m, int n)
{
    int i=0, temp, x, a=m, b=n;
    while(m%2==0 && n%2==0)
    {
        m/=2;
        n/=2;
        i++;
    }
    if(m<n)
    {
       temp = m;
       m = n;
       n = temp;
    }
    while(x)
    {
        x = m-n;
        m = (x>n)?x:n;
        n = (x>n)?n:x;
        if(n==(m-n))
           break;
    }
    if(i==0)

       printf("最大公约数为： %d  最小公倍数为： %d", n, a*b/n);
     else
     {
        temp = (int)(pow(2,i)*n);
        printf("最大公约数为： %d  最小公倍数为： %d", temp, a*b/temp);
    }
```

```
}
```

20. 输入 4 个整数，找出其中最大的数。用函数的嵌套调用来处理。

```
#include <stdio.h>
//题目要求函数嵌套
int max_t(int m, int n);
int max_f(int a, int b, int c, int d);

int main()
{
    int a, b, c, d;
    printf("请输入四个整数：");
    scanf("%d %d %d %d",&a, &b, &c, &d);

    printf("四个整数中最大值为：%d", max_f(a, b, c, d));
    return 0;
}

int max_f(int a, int b, int c, int d)
{
     if(max_t(a,b)>max_t(c,d))
       return max_t(a,b);
     else
       return max_t(c,d);
}
int max_t(int m, int n)
{
   if(m>n)
   return m;
   else
   return n;

}
```

21. 用递归方法求 5！（深刻体会递归含义，理解思想）

```
#include <stdio.h>
int factorial(int n);
int main()
{
    int a;
    printf("请输入一个整数：");
    scanf("%d", &a);
```

```
    if(a>=0)
    printf("该数的阶乘为：%d", factorial(a));
    else
    factorial(a);

    return 0;
}

int factorial(int n)
{
     int f;
     if(n<0)
     printf("输入错误！n 应该大于等于 0！");
     else if(n==0||n==1)
     f=1;
     else f = n*factorial(n-1);
     return f;
}
```

22. 编写一个函数，用来分别求数组 score_1（有 5 个元素）和数组 score_2（有 10 个元素）各元素的平均值。

方法一：数组

```
#include "stdio.h"
int main()
{
   float f(float score[],int n);
   float score_1[5],score_2[10],aver_1,aver_2;
   int i;
   for(i=0;i<5;i++)
    scanf("%f",&score_1[i]);
   for(i=0;i<10;i++)
    scanf("%f",&score_2[i]);
   aver_1=f(score_1,5);
   aver_2=f(score_2,10);
   printf("aver_1=%f,aver_2=%f\n",aver_1,aver_2);

   return 0;
}
float f(float score[],int n)
{
     int i;
```

```
    float aver,sum=0;
    for(i=0;i<n;i++)
      sum=sum+score[i];
     aver=sum/n;
     return(aver);
}
```

方法二：指针

```
#include <stdio.h>
float mean(float *p, int n);
int main()
{
    int i;
    float score_1[5], score_2[10];

    printf("请输入数组一：\n");
    for(i=0; i<5; i++)
    {
      scanf("%f", &score_1[i]);
    }
    printf("数组一的平均值为：%.1f", mean(score_1, 5));

    printf("\n");
    printf("请输入数组二：\n");
    for(i=0; i<10; i++)
    {
      scanf("%f", &score_2[i]);
    }
    printf("数组一的平均值为：%f", mean(score_2, 10));

    return 0;
}

float mean(float *p, int n)
{
   float sum = 0;
   for(int i=0; i<n; i++)
   {
     sum += *p;
     p++;
   }
    return sum/n;
}
```

23. 将一个数组中的值按逆序重新存放。例如，原来顺序为 8, 6, 5, 4, 1 改后成为 1, 4, 5, 6, 8。

```
#include <stdio.h>
void reverse_arr(int a[], int n);
int main()
{
    int i, a[5], b[8];
    printf("请输入几个数：");
    for(i=0; i<5; i++)
    {
      scanf("%d", &a[i]);
    }

    for(i=0; i<8; i++)
    {
      scanf("%d", &b[i]);
    }

    reverse_arr(a, 5);
    reverse_arr(b, 8);

    printf("a 数组逆序为：");
    for(i=0; i<5; i++)
    {
      printf("%d ", a[i]);
    }

    printf("\n");
    printf("b 数组逆序为：");
    for(i=0; i<8; i++)
    {
      printf("%d ", b[i]);
    }

    return 0;
}

void reverse_arr(int a[], int n)
{
   int i, temp;
   for(i=0; i<n/2; i++)
   {
     temp = a[i];
     a[i] = a[n-i-1];
```

```
        a[n-i-1] = temp;
    }

}
```

24. 用递归法将一个整数 n 转换成字符串。例如，输入 483，应输出字符串“483”，n 的位数不确定，可以是任意位数的整数。

```
#include <stdio.h>
//递归法
void convert(int n);
void convert1(int n);//非递归逆序
int main()
{
    int a;
    printf("请输入一个数字：");
    scanf("%d", &a);
    printf("输出字符为：");
    if(a<0)
    {
        putchar('-');
        putchar(' ');
        a = -a;
    }
    convert1(a);
    printf("\n");

    return 0;
}

void convert(int n)
{
    int i;
    if((i=n/10)!=0)
        convert(i);
    putchar(n%10+'0');
    putchar(32);
}

void convert1(int n)
{
    int t;
    while(n>0)
    {
```

```
            t = n%10;
            putchar(t+'0');
            putchar(32);
            n = n/10;
        }

}
```

25. 用指针实现输入 3 个整数并按从小到大的顺序输出。

```
#include <stdio.h>
void swap(int *p, int *q);
void sort(int *p, int *q, int *r);
int main()
{
    int a, b, c;
    int *p1, *p2, *p3;
    printf("请输入三个整数：");
    scanf("%d %d %d", &a, &b, &c);
    p1 = &a;
    p2 = &b;
    p3 = &c;
    printf("从小到大排序结果为：");
    sort(p1, p2, p3);
    printf("%d %d %d", a, b, c);

    return 0;
}
void sort(int *p, int *q, int *r)
{
      if(*p>*q)
         swap(p, q);
      if(*p>*r)
         swap(p, r);
      if(*q>*r)
         swap(q, r);
}
void swap(int *p, int *q)
{
      int temp;
      temp = *p;
      *p = *q;
      *q = temp;
}
```

26. 用指针实现输入 3 个整数并按从大到小的顺序输出。

```
#include <stdio.h>
void swap(int *p, int *q);
void sort(int *p, int *q, int *r);
int main()
{
   int a, b, c;
   int *p1, *p2, *p3;
   printf("请输入三个整数：");
   scanf("%d %d %d", &a, &b, &c);
   p1 = &a;
   p2 = &b;
   p3 = &c;
   printf("从小到大排序结果为：");
   sort(p1, p2, p3);
   printf("%d %d %d", a, b, c);

   return 0;
}
void sort(int *p, int *q, int *r)
{
     if(*p<*q)
        swap(p, q);
     if(*p<*r)
        swap(p, r);
     if(*q<*r)
        swap(q, r);
}
void swap(int *p, int *q)
{
     int temp;
     temp = *p;
     *p = *q;
     *q = temp;
}
```

27. 用指针实现将数组 b[10]中的元素按逆序存放。

```
#include <stdio.h>
void reverse(int a[], int n);
int main()
{
   int i, b[10];
   printf("请输入十个数：");
   for(i=0; i<10; i++)
   {
      scanf("%d", &b[i]);
   }
   reverse(b, 10);

    printf("逆序后数组为：");
    for(i=0; i<10; i++)
   {
      printf("%d ", b[i]);
   }

   return 0;
}

void reverse(int a[], int n)
{
     int temp, i, *p, *q;
     for(i=0; i<n/2; i++)
     {
          p=a+i;
          temp = *p;
          q = a+n-1-i;
          *p = *q;
          *q = temp;
     }

}
```

28. 用冒泡法实现对 10 个整数按从大到小的顺序排序输出（要求用指针实现）。

```
#include <stdio.h>

void swap(int *p, int *q);
```

```
int main()
{
   int i, j, b[10], *p;
   printf("请输入十个数：");
   for(i=0; i<10; i++)
   {
      scanf("%d", &b[i]);
   }
   p = b;
    for(i=0; i<10; i++)
    {
       for(j=9; j>i; j--)
       {
            if(b[j]>b[j-1])//*(p+j)>*(p+j-1)
            swap(p+j, p+j-1);
       }

    }
    printf("冒泡排序后数组为：");
    for(i=0; i<10; i++)
   {
      printf("%d ", b[i]);
   }
   return 0;
}
void swap(int *p, int *q)
{
     int temp;
     temp = *p;
     *p = *q;
```

```
    *q = temp;
}
```

29. 选择法实现对 10 个整数按从大到小的顺序排序输出（要求用指针实现）。

```
#include <stdio.h>
void SelectSort(int *p, int n);
 //选择排序
int main()
{
   int i, j, temp, max, b[10], *p;
   printf("请输入十个数：");
   for(i=0; i<10; i++)
   {
      scanf("%d", &b[i]);
   }
   p = b;

   SelectSort(p, 10);

   printf("冒泡排序后数组为：");
   for(i=0; i<10; i++)
   {
      printf("%d ", b[i]);
   }

   return 0;
}

void SelectSort(int *p, int n)
{
   int i, j, k, temp;
   for(i=0; i<n; i++)
   {
     k = i;
     for(j = i+1; j<n; j++)
         if(*(p+i)<*(p+j))
             k = j;
     if(k!=i)
      {
        temp = *(p+i);
```

```
            *(p+i) = *(p+k);
            *(p+k) = temp;
        }
    }
}
```

30. 用选择法实现对10个整数按从小到大的顺序排序输出（要求用指针实现）。

上面排序算法的小于号改成大于号

```
#include <stdio.h>
void SelectSort(int *p, int n);
int main()
{
    int i, j, temp, max, b[10], *p;
    printf("请输入十个数：");
    for(i=0; i<10; i++)
    {
        scanf("%d", &b[i]);
    }
    p = b;
    SelectSort(p, 10);

    printf("冒泡排序后数组为：");
    for(i=0; i<10; i++)
    {
        printf("%d ", b[i]);
    }
    return 0;
}

void SelectSort(int *p, int n)
{
```

```
    int i, j, k, temp;
    for(i=0; i<n; i++)
    {
      k = i;
      for(j = i+1; j<n; j++)
          if(*(p+i)>*(p+j))
              k = j;
      if(k!=i)
       {
         temp = *(p+i);
         *(p+i) = *(p+k);
         *(p+k) = temp;
       }
    }
}
```

31. 定义一个结构体变量（包括年、月、日）。计算当天是本年中的第几天，注意闰年问题。

```
#include <stdio.h>
struct{
      int year;
      int mouth;
      int day;
}date;
/*判断任意年份是否为闰年，需要满足以下条件中的任意一个：
① 该年份能被 4 整除同时不能被 100 整除；
② 该年份能被 400 整除。*/
int main()
{
   int days;
   printf("请输入年月日：");
   scanf("%d %d %d", &date.year, &date.mouth, &date.day);
   switch(date.mouth)
   {
```

```
    case 1: days = date.day;    break;
    case 2: days = date.day+31; break;
    case 3: days = date.day+31+28; break;
    case 4: days = date.day+31+28+31; break;
    case 5: days = date.day+31+28+31+30; break;
    case 6: days = date.day+31+28+31+30+31; break;
    case 7: days = date.day+31+28+31+30+31+30; break;
    case 8: days = date.day+31+28+31+30+31+30+31; break;
    case 9: days = date.day+31+28+31+30+31+30+31+31; break;
    case 10: days = date.day+31+28+31+30+31+30+31+31+30; break;
    case 11: days = date.day+31+28+31+30+31+30+31+31+30+31; break;
    case 12: days = date.day+31+28+31+30+31+30+31+31+30+31+30; break;
  }
  if((date.year%4==0 && date.year%100!=0 || date.year%400==0)&&date.mouth>=3)
        days++;

  printf("%d 年%d 月%d 日是该年的第%d 天！ ", date.year, date.mouth, date.day, days);
  return 0;
}
```

32. 设计候选人得票统计程序，要求有 4 个侯选人（分别是 Zhang 、Wang 、Li、Zhao），选民每次输入一个被选人的姓名，最后统计出各人的得票结果。

方法一：结构体

```
#include <string.h>
#include <stdio.h>
struct person
  {char name[20];
   int count;
  }leader[4]={"zhang",0,"wang",0,"li",0,"zhao",0};

int main()
{
  int i,j;
  char leader_name[20];
  for (i=1;i<=10;i++)
    {scanf("%s",leader_name);
       for(j=0;j<4;j++)
         if(strcmp(leader_name,leader[j].name)==0) leader[j].count++;
     }
  printf("\nResoult:\n");
```

```
    for(i=0;i<4;i++)
        printf("%5s:%d\n",leader[i].name,leader[i].count);
    return 0;
}
```

方法二：

```
#include <stdio.h>
#include <string.h>
int main()
{
    char  name[20],  name1[20]="zhang",  name2[20]="wang",  name3[20]="li",
name4[20]="zhao";
    int   cand1=0, cand2=0, cand3=0, cand4=0;

    printf("请各位选民输入被选人姓名：\n");
    while(scanf("%s",name)!=EOF){   //当按下 ctr+z 终止输入时，scanf 会返回
EOF(-1)\n
        if(strcmp(name, name1)==0)
            {
                cand1++;
                printf("候选人 zhang 的得票为：%d\n", cand1);
                printf("候选人 wang 的得票为：%d\n", cand2);
                printf("候选人 li 的得票为：%d\n", cand3);
                printf("候选人 zhao 的得票为：%d\n", cand4);
            }
        if(strcmp(name, name2)==0)
            {
                cand2++;
                printf("候选人 zhang 的得票为：%d\n", cand1);
                printf("候选人 wang 的得票为：%d\n", cand2);
                printf("候选人 li 的得票为：%d\n", cand3);
                printf("候选人 zhao 的得票为：%d\n", cand4);
            }
        if(strcmp(name, name3)==0)
            {
                cand3++;
                printf("候选人 zhang 的得票为：%d\n", cand1);
                printf("候选人 wang 的得票为：%d\n", cand2);
                printf("候选人 li 的得票为：%d\n", cand3);
                printf("候选人 zhao 的得票为：%d\n", cand4);
            }
        if(strcmp(name, name4)==0)
            {
                cand4++;
```

```
            printf("候选人 zhang 的得票为：%d\n", cand1);
            printf("候选人 wang 的得票为：%d\n", cand2);
            printf("候选人 li 的得票为：%d\n", cand3);
            printf("候选人 zhao 的得票为：%d\n", cand4);
        }
    }
    return 0;
}
```

33. 定义一个包括学号、姓名、年龄的学生结构体，要求实现三个学生信息的输入输出，并输出平均年龄

```
#include <stdio.h>
struct student{
    int sno;
    char name[20];
    int age;
};
int main()
{
    int i;
    float ave=0;
    student st[3];
    printf("请输入三个学生的学号姓名年龄：\n");
    for(i=0; i<3; i++)
    {
      scanf("%d %s %d", &st[i].sno, st[i].name, &st[i].age);
      ave = ave + st[i].age;
    }
    ave = ave/3;

    printf("三个学生的信息为：\n");
     for(i=0; i<3; i++)
    {
      printf("%d %s %d\n", st[i].sno, st[i].name, st[i].age);
    }
    printf("三个学生的平均年龄为：%f", ave);

    return 0;
}
```

或者：

```
#include <stdio.h>
#define N 3
struct student
{ char num[6];
   char name[20];
   int age;
} stu[N];
int main()
{
   int i,aver=0;
   for(i=0;i<N;i++)
     {
       printf("input scores of student %d:\n",i+1);
        printf("NO.:");
        scanf("%s",stu[i].num);
        printf("name:");
        scanf("%s",stu[i].name);
        printf("age :");
       scanf("%d",&stu[i].age);
     }
   printf("     NO.          name     age\n");
   for (i=0;i<N;i++)
      {
          printf("%5s%10s%9d",stu[i].num,stu[i].name,stu[i].age);
             printf("\n");
      }
    for(i=0;i<N;i++)
       aver=aver+stu[i].age;
    aver=aver/3;
    printf("aver=%d",aver);
    return 0;
}
```

34. 定义一个包括教师编号、姓名、工资的教师结构体，要求实现三个教师信息的输入输出，并输出平均工资。

```
#include <stdio.h>
struct teacher{
      int tno;
      char name[20];
```

```
    int salar;
};
int main()
{
   int i;
   float ave=0;
   teacher te[3];
   printf("请输入三个教师的编号-姓名-工资：\n");
   for(i=0; i<3; i++)
   {
     printf("NO:");
     scanf("%d", &te[i].tno);
     printf("name:");
     scanf("%s", te[i].name);
     printf("salary:");
     scanf("%d", &te[i].salar);

     ave = ave + te[i].salar;
   }
   ave = ave/3;

   printf("三个老师的信息为：\n");
   printf("编号    姓名    工资\n");
    for(i=0; i<3; i++)
   {
     printf("%d    %s    %d\n", te[i].tno, te[i].name, te[i].salar);
   }
   printf("三个老师的平均工资为：%f", ave);

   return 0;
}
```

35. 定义一个包括职员号、姓名、工资的职员结构体，要求实现三个职员信息的输入输出，并输出平均工资。

```
#include <stdio.h>
struct worker{
    int wno;
    char name[20];
    int salar;
```

```
};
int main()
{
   int i;
   float ave=0;
   worker wo[3];
   printf("请输入三个职员的编号-姓名-工资：\n");
   for(i=0; i<3; i++)
   {
     printf("NO:");
     scanf("%d", &wo[i].wno);
     printf("name:");
     scanf("%s", wo[i].name);
     printf("salary:");
     scanf("%d", &wo[i].salar);

     ave = ave + wo[i].salar;
   }
   ave = ave/3;

   printf("三个职员的信息为：\n");
   printf("编号    姓名    工资\n");
    for(i=0; i<3; i++)
   {
     printf("%d     %s       %d\n", wo[i].wno, wo[i].name, wo[i].salar);
   }
   printf("三个职员的平均工资为：%5.1f", ave);

   return 0;
}
```

36. 建立动态数组，输入 5 个学生的成绩，另外用一个函数检查其中有无低于 60 分的，输出不合格的成绩。

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

```
int check(int *p, int n);
int main()
{
   int i, *score;
   score = (int *)malloc(5*sizeof(int));
   printf("请输入五个学生成绩：");
   for(i=0; i<5; i++)
    scanf("%d", score+i);

   if(check(score, 5))
   {
     printf("存在不合格成绩，不合格成绩为：");
     for(i=0; i<5; i++)
     {
          if(*(score+i)<60)
          printf("%d ", *(score+i));
     }
   }
   else
   printf("无不合格成绩！");
   return 0;
}

int check(int *p, int n)
{
     int i;
     for(i=0; i<n; i++)
     {
          if(*(p+i)<60)
          return 1;
     }
     return 0;
}
```

或者

```
#include <stdio.h>
#include <stdlib.h>
int main()
{ void check(int *);
   int *p1,i;
   void *p2;
   p2=malloc(5*sizeof(int));
   p1=(int *)p2;
   for(i=0;i<5;i++)
       scanf("%d",p1+i);
```

```
    check(p1);
    free(p2);

    return 0;
}

void check(int *p)
{
    int i;
    printf("They are fail:");
    for(i=0;i<5;i++)
      if (p[i]<60) printf("%d ",p[i]);
    printf("\n");
}
```

37. 写一函数，求一个字符串的长度。在 main 函数中输入字符串，并输出其长度（要求用指针实现）。

```
#include <stdio.h>
int char_long(char *p);

int main()
{
    char ch[20];
    printf("请输入一个字符串： ");
    scanf("%s", ch);

    printf("该字符串长度为： %d", char_long(ch));

    return 0;
}
int char_long(char *p)
{
     int i, clong=0;
     while(*p!='\0')
     {
         clong++;
         p++;
     }
     return clong;
}
```

38. 有一个一维数组 score，内放 10 个学生成绩，用一个函数求平均成绩，并将 10 个成绩中不及格（小于 60）的成绩和该成绩在数组中的序号输出。

```
#include <stdio.h>
void avg_print(int a[], int n);
int main()
{
   int score[] = {88, 99, 56, 45, 89, 48, 36, 51,77, 58};

   avg_print(score, 10);

   return 0;
}

void avg_print(int a[], int n)
{
    int i;
    float avg;
    for(i=0; i<n; i++)
    {
     avg = avg+a[i];
     if(a[i]<60)
     {
        printf("不合格成绩为：%d,", a[i]);
        printf("该成绩的序号为：%d\n", i+1);
     }
    }

    printf("学生的平均成绩为：%5.1f", avg/n);
}
```

或者（下面答案缺少了求平均成绩）

```
#include "stdio.h"
int main()
{
   void fail(float score[],int n);
   float score[10];
   int i;
   for(i=0;i<10;i++)
      scanf("%f",&score[i]);
   fail(score,10);
   return 0;
}
 void fail(float score[],int n)
```

```
{
    int i;
    for(i=0;i<n;i++)
     if(score[i]<60)
          printf("成绩为%f,序号为%d\n",score[i],i+1);
}
```

39. 有一个一维数组内放 10 个数，设计函数，求出 10 个数中的最大值、最小值、平均值。

```
#include "stdio.h"
void print_mm(int a[], int n);

int main()
{
   int i, a[10];
   printf("请输入十个数：");
   for(i=0; i<10; i++)
     {
      scanf("%d", &a[i]);
     }
   print_mm(a, 10);

   return 0;
}
void print_mm(int a[], int n)
{
     int i, max, min;
     float avg = 0;
     max = a[0];
     min = a[0];
     for(i=0; i<n; i++)
     {
          avg += a[i];
          if(a[i]>max)
               max = a[i];
          if(a[i]<min)
               min = a[i];
     }
     printf("这十个数的最大值为%d,最小值为%d，平均值为%5.1f", max, min, avg/n);
```

```
}
```
或者
```
#include "stdio.h"
int main()
{
   int a[10],i,max,min,sum=0;
     float average;
   printf("input 10 number:\n");
   for(i=0;i<10;i++)
      scanf("%d",&a[i]);
   max=a[0];
   for(i=0;i<10;i++)
     {
        sum=sum+a[i];
        if(a[i]>max)
            max=a[i];
         }
    average=sum/10.0;
         min=a[0];
      for(i=0;i<10;i++)
          {
                if(a[i]<min)
              min=a[i];
          }
       printf("max=%d,average=%f,min=%d",max,average,min);

       return 0;
}
```

40. 找出一个 2 维数组中的鞍点，即该位置上的元素在该行上最大、在该列上最小。

```
#include <stdio.h>

int main()
{
   int a[2][3]={{1,2,3},{4,5,6}};
   int i, j, max, min, ki, kj;

   for(i=0; i<2; i++)
   {
     max = a[i][0];
     for(j=0; j<3; j++)
```

```
    {
        if(a[i][j]>max)
        {
            max = a[i][j];
            ki = i;
            kj = j;
        }
    }
    min = max;
    for(j=0; j<2; j++)
    {
        if(a[j][kj]<min)
            min = a[j][kj];
    }
    if(max==min)
       printf("该数组鞍点为%d, 在%d 行%d 列", max, ki+1, kj+1);
  }

  return 0;
}
```

你电答案解法如下，（前面标注或者的都是你电解法）：

```
#include "stdio.h"
void main()
{
    int a[4][5];
    int i,j,k,m;
    for(i=0;i<4;i++)
      for(j=0;j<5;j++)
       scanf("%d",&a[i][j]);
    for(i=0;i<4;i++)
      for(j=0;j<5;j++)
       for(m=0;m<4;m++)
        if(a[i][j]<a[m][j])
          for(k=0;k<5;k++)
            if(a[i][j]>a[i][k])
             for(k=0;k<5;k++)
               if(a[i][j]>a[i][k]) continue;
      printf("i=%d,j=%d",i,j);
     }
```

40. 输入 8 个学生 4 门课的成绩，分别用函数实现如下功能：

（1） 计算每个学生平均分；

（2） 计算每门课的平均分；

（3） 找出 32 个分数中最高分所对应的学生和课程。

方法一：（结构体）

```
#include <stdio.h>
struct sc{
   float c1;
   float c2;
   float c3;
   float c4;
};
void sc_avg(sc s[], int n);
float s_avg(sc s);
float highest(sc s[], int n);
int st, cour;

int main()
{
   int i;
   float hest;
   sc s[8];
   printf("请输入 8 个学生四门课成绩：\n");
   printf("c1 c2 c3 c4\n");
   for(i=0; i<8; i++)
   {
     scanf("%f %f %f %f", &s[i].c1, &s[i].c2, &s[i].c3, &s[i].c4);
   }

   printf("8 个学生平均分如下：\n");
   for(i=0; i<8; i++)
   {
     printf("%5.1f\n", s_avg(s[i]));
   }
   sc_avg(s, 8);
   hest = highest(s, 8);
   printf("所有成绩中最高分为：%5.1f，是第%d 个学生 course%d", hest, st,
cour);
   return 0;
}

float s_avg(sc s)
{
   float avg;
```

```
    avg = (s.c1 + s.c2 + s.c3 + s.c4)/4;
    return avg;
}

void sc_avg(sc s[], int n)
{
    int i;
    float avg1, avg2, avg3, avg4;
    for(i=0; i<n; i++)
    {
        avg1 += s[i].c1;
        avg2 += s[i].c2;
        avg3 += s[i].c3;
        avg4 += s[i].c4;
    }

    printf("course1 平均分为%5.1f\n", avg1/n);
    printf("course2 平均分为%5.1f\n", avg2/n);
    printf("course3 平均分为%5.1f\n", avg3/n);
    printf("course4 平均分为%5.1f\n", avg4/n);
}

float highest(sc s[], int n)
{
      int i;
      float max=s[0].c1;
      for(i=0; i<n; i++)
      {
            if(s[i].c1>max)
            {
               max = s[i].c1;
               st = i+1;
               cour = 1;
            }

            if(s[i].c2>max)
            {
                max = s[i].c2;
               st = i+1;
               cour = 2;
            }

            if(s[i].c3>max)
            {
```

```
            max = s[i].c3;
            st = i+1;
            cour = 3;
          }

          if(s[i].c4>max)
          {
            max = s[i].c4;
            st = i+1;
            cour = 4;
          }

      }
      return max;
}
```

方法二：二维数组

```
#include <stdio.h>
#define N 8
#define M 4
float score[N][M];
float a_stu[N],a_cour[M];
int r,c;
int main()
{ int i,j;
   float h;
   float highest();
   void input_stu(void);
   void aver_stu(void);
   void aver_cour(void);
   input_stu();
   aver_stu();
   aver_cour();
   printf("\n   NO.      cour1    cour2    cour3    cour4   aver\n");
   for(i=0;i<N;i++)
    {printf("\n NO %2d ",i+1);
     for(j=0;j<M;j++)
        printf("%8.2f",score[i][j]);
     printf("%8.2f\n",a_stu[i]);
    }
   printf("\naverage:");
   for (j=0;j<M;j++)
      printf("%8.2f",a_cour[j]);
   printf("\n");
```

```
    h=highest();
    printf("highest:%7.2f      NO. %2d      course %2d\n",h,r,c);

    return 0;
}
void input_stu(void)
  {int i,j;
    for (i=0;i<N;i++)
      {printf("\ninput score of student%2d:\n",i+1);
        for (j=0;j<M;j++)
           scanf("%f",&score[i][j]);
      }
  }
void aver_stu(void)
  {int i,j;
    float s;
    for (i=0;i<N;i++)
      {for (j=0,s=0;j<M;j++)
           s+=score[i][j];
        a_stu[i]=s/5.0;
      }
  }
void aver_cour(void)
  {int i,j;
    float s;
    for (j=0;j<M;j++)
       {s=0;
         for (i=0;i<N;i++)
            s+=score[i][j];
         a_cour[j]=s/(float)N;
       }
  }
float highest()
  {float high;
    int i,j;
    high=score[0][0];
    for (i=0;i<N;i++)
       for (j=0;j<M;j++)
          if (score[i][j]>high)
      {high=score[i][j];
        r=i+1;
        c=j+1;
      }
    return(high);
```

```
}
```

41. 用一个函数实现将一行字符串中的最长的单词输出。此行字符串从主函数传递给该函数。

```
#include <stdio.h>
void print_longest(char str[]);
int main()
{
   int i;
   char str[50];
   printf("请输入字符串：");
   gets(str);
   print_longest(str);

   return 0;
}
void print_longest(char str[])
{
    int i, k, lg=0, max=0;
    while(str[i]!='\0')
    {
      if(('a'<str[i] && str[i]<'z') || ('A'<str[i] && str[i]<'Z'))
          {
            lg++;
            if(lg>max)
            {
             max = lg;
             k = i;
            }
```

```
            }
        else
            lg=0;
        i++;
    }
    printf("该字符串最长单词长度为：%d\n", max);
    printf("该字符串最长单词为：\n");
    for(i=k-max+1; i<=k; i++)
    {
      printf("%c", str[i]);
    }
}
```

或者

```
#include <stdio.h>
#include <string.h>
void main()
{int alphabetic(char);
 int longest(char []);
 int i;
 char line[100];
 printf("input one line:\n");
 gets(line);
 printf("The longest word is :");
 for (i=longest(line);alphabetic(line[i]);i++)
    printf("%c",line[i]);
 printf("\n");
}
int alphabetic(char c)
{if ((c>='a' && c<='z')||(c>='A'&&c<='z'))
   return(1);
 else
   return(0);
}
int longest(char string[])
{int len=0,i,length=0,flag=1,place=0,point;
 for (i=0;i<=strlen(string);i++)
    if (alphabetic(string[i]))
       if (flag)
```

```
            {point=i;
             flag=0;
            }
          else
            len++;
      else
         {flag=1;
          if (len>=length)
       {length=len;
        place=point;
        len=0;
       }
         }
  return(place);
}
```

42. 输出金字塔图案（要求用循环实现）

```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```

```
#include <stdio.h>
int main()
{
   int i, j, k;
   for(i=0; i<5; i++)
   {
     for(j=0; j<4-i; j++)
        printf(" ");
     for(k=0; k<2*i+1; k++)
        printf("*");
     printf("\n");
   }

   for(i=0; i<4; i++)
   {
     for(j=0; j<i+1; j++)
        printf(" ");
     for(k=0; k<7-2*i; k++)
        printf("*");
```

```
        printf("\n");
    }

    return 0;
}
```

或者

```
#include <stdio.h>
void main()
  {int i,j,k;
    for (i=0;i<=3;i++)
      {for (j=0;j<=2-i;j++)
          printf(" ");
       for (k=0;k<=2*i;k++)
          printf("*");
       printf("\n");
      }
    for (i=0;i<=2;i++)
      {for (j=0;j<=i;j++)
          printf(" ");
       for (k=0;k<=4-2*i;k++)
          printf("*");
       printf("\n");
      }
    }
```

43. 要有一个已排好序的数组，求输入一个数后，按原来排序的规律将她插入数组中

```
#include <stdio.h>

int main()
{
    int a[7] = {1, 2, 3, 4, 5, 6};
    int i, t, k=6;
    printf("请输入一个数： ");
    scanf("%d", &t);

    for(i=5; i>=0; i--)
    {
        if(a[i]>t)
        {
          a[i+1] = a[i];
```

```
            k = i;
        }

    }
    a[k] = t;
    printf("插入该数后的数组为：");
    for(i=0; i<7; i++)
    {
      printf("%d ", a[i]);
    }

    return 0;
}
```

或者

```
#include <stdio.h>
int main()
{
      int a[10];
      int b[11];
      int i,m,j;
      printf("请输入一个有序数组：");
      for(i=0;i<10;i++)
      scanf("%d",&a[i]);
      printf("请输入一个任意的整数：");
      scanf("%d",&m);
      for(i=0;i<10;i++)
              if(m<a[i])break;
      for(j=0;j<11;j++)
         {
         if(i>j)
         b[j]=a[j];
         else if(i<j)
         b[j]=a[j-1];
             else b[j]=m;
         }
        printf("该数组的重新排序为:");
        for(j=0;j<11;j++)
         printf("%d    ",b[j]);

    return 0;
}
```

44. 写一个判断素数的函数，在主函数输入一个整数，输出是否素数的信息。

```
#include <stdio.h>
int s_standard(int n);
int panduan(int n);//两个均可第一个效率较高
int main()
{
   int a;
   printf("请输入一个数： ");
   scanf("%d", &a);
   if(s_standard(a)==1)
      printf("%d 是素数", a);
   else
    printf("%d 不是素数", a);

   return 0;
}
int s_standard(int n)
{
      int i;
      for(i=2; i*i<n; i++)
      {
          if(n%i==0)
          return 0;
      }
      if(i*i>=n)
       return 1;

}
int panduan(int n)
{
 int i,temp;
 temp=1;
 for(i=2;i<n/2&&temp==1;i++)
       if(n%i==0)
            temp=0;
      return (temp);
}
```

45. 有一篇文章，共有 4 行文章，每行有 60 个字符。要求分别统计出其中英文大写字符、小写字母、数字、空格以及其他字符的个数。

```
#include <stdio.h>

int main()
{
```

```
    int i=0, j=0, dx=0, xx=0, numb=0, kg=0, other=0;
    char text[4][60], c;

    for(i=0; i<4; i++)
    {
      printf("请输入第%d 行：\n", i+1);
      gets(text[i]);
      j=0;
      while((c=text[i][j++])!='\0')
        {
          if(c<='Z'&& c>='A')
             dx++;
          else if(c<='z' && c>='a')
             xx++;
          else if(c>='0' && c<='9')
             numb++;
          else if(c==' ')
             kg++;
          else other++;
        }
    }

    printf("这篇文章有：\n");
    printf("大写字母有%d 个\n", dx);
    printf("小写字母%d 个\n", xx);
    printf("数字%d 个\n", numb);
    printf("空格%d 个\n", kg);
    printf("其他字符%d 个\n", other);

    return 0;
}
```

或者

```
#include <stdio.h>
void main()
  {int i,j,upp,low,dig,spa,oth;
    char text[4][60];
    upp=low=dig=spa=oth=0;
    for (i=0;i<4;i++)
      { printf("please input line %d:\n",i+1);
        gets(text[i]);
        for (j=0;j<60 && text[i][j]!='\0';j++)
          {if (text[i][j]>='A'&& text[i][j]<='Z')
             upp++;
```

```
            else if (text[i][j]>='a' && text[i][j]<='z')
               low++;
            else if (text[i][j]>='0' && text[i][j]<='9')
               dig++;
            else if (text[i][j]==' ')
              spa++;
            else
              oth++;
        }
     }
        printf("\nupper case: %d\n",upp);
        printf("lower case: %d\n",low);
        printf("digit       : %d\n",dig);
        printf("space       : %d\n",spa);
        printf("other       : %d\n",oth);
}
```

46. 有一行电文，已按下面规律译成密码：

A→Z　　a→z

B→Y　　b→y

C→X　　c→x

即第 1 个字母变成第 26 个字母，第 i 个字母变成第（26 - i + 1）个字母。非字母符号不变。要求编程序将密码译回原文，并输出密码和原文。

```
#include <stdio.h>
int main()
{
  int j, n;
  char str[50], trans[50];
  printf("请输入电文: ");
  gets(str);
  printf("原文: %s\n", str);
  j=0;
  while(str[j]!='\0')
  {
    if(str[j]>='A' && str[j]<='Z')
      trans[j] = 155-str[j];
    else if(str[j]>='a' && str[j]<='z')
      trans[j] = 219-str[j];
    else
      trans[j] = str[j];
    j++;
  }
```

```
    printf("译文: %s\n", trans);

    return 0;
}
```

或者

```
#include <stdio.h>
int main()
{
   int j,n;
   char ch[80],tran[80];
   printf("input cipher code:");
   gets(ch);
   printf("\ncipher code    :%s",ch);
   j=0;
   while (ch[j]!='\0')
   { if ((ch[j]>='A') && (ch[j]<='Z'))
         tran[j]=155-ch[j];
     else if ((ch[j]>='a') && (ch[j]<='z'))
         tran[j]=219-ch[j];
     else
         tran[j]=ch[j];
     j++;
   }
   n=j;
   printf("\noriginal text:");
   for (j=0;j<n;j++)
     putchar(tran[j]);
   printf("\n");
   return 0;
 }
```

47. 有 n 个人围成一圈，顺序排号。从第一个人开始报数（从 1 到 3 报数），凡报到 3 的人退出圈子，问最后留下来的是原来第几号的那位（要求用指针知识实现）。

```
#include <stdio.h>
#include <stdlib.h>
int main()
```

```
{
  int n, i, k, m, *p;
  printf("请输入人数: ");
  scanf("%d", &n);
  p=(int*)malloc(n*sizeof(int));//成电答案用的数组
  for(i=0; i<n; i++)
    *(p+i) =i+1;
  k=0;
  m=0;
  i=0;
  while(m<n-1)
  {
    if(*(p+i)!=0)
       k++;
    if(k==3)
      {
       *(p+i)=0;
       k=0;
       m++;
      }
    i++;
    if(i==n)
       i=0;
  }
  while(*p==0)
    p++;
  printf("最后一个人是起初第%d 号!\n", *p);
  return 0;
}
```

48. 有 10 个学生，每个学生的数据包括学号、姓名、3 门课程的成绩，从键盘输入 10 个学生数据，要求输出 3 门课程总平均成绩，以及最高分的学生的数据（包括学号、姓名、3 门课程成绩、平均分数）（用结构体）。

```
#include <stdio.h>
#define N 10
struct student{
    int sno;
    char name[20];
    float c1;
    float c2;
    float c3;
```

```
    float avg;
};
int main()
{
   int i, maxi;
   float sum[N], average, max=0;
   student stu[N];
   //总分最高即平均分最高
   printf("请输入十个学生数据：学号-姓名-三门课成绩\n");
   printf("sno   name   cour1 cour2 cour3\n");
   for(i=0; i<N; i++)
   {
     scanf("%d %s %f %f %f", &stu[i].sno, stu[i].name, &stu[i].c1, &stu[i].c2,
&stu[i].c3);
     stu[i].avg = (stu[i].c1 + stu[i].c2 + stu[i].c3)/3.0;
     average +=  stu[i].avg;
     sum[i] = stu[i].c1 + stu[i].c2 + stu[i].c3;
     if(sum[i]>max)
     {
        max= sum[i];
        maxi = i;
     }

   }
   average = average/N;

   printf("输入的学生信息如下：\n");
   printf("sno   name c1 c2 c3 avg\n");
   for(i=0; i<N; i++)
     printf("%d %s %5.1f %5.1f %5.1f %5.1f\n", stu[i].sno, stu[i].name, stu[i].c1,
stu[i].c2, stu[i].c3, stu[i].avg);

   printf("总平均成绩为：%5.1f\n", average);

   printf("最高分学生数据为：\n");
   printf("学号%d 姓名%s \n", stu[maxi].sno, stu[maxi].name);
   printf("分数:%5.1f, %5.1f, %5.1f  平均分:%5.1f", stu[maxi].c1, stu[maxi].c2,
stu[maxi].c3, stu[maxi].avg);

   return 0;
}
或者（讲真的这方法太磨叽）
#include <stdio.h>
#define N 10
```

```
struct student
{ char num[6];
   char name[8];
   float score[3];
   float avr;
} stu[N];
void main()
{ int i,j,maxi;
   float sum,max,average;
   for (i=0;i<N;i++)
      {printf("input scores of student %d:\n",i+1);
       printf("NO.:");
       scanf("%s",stu[i].num);
       printf("name:");
       scanf("%s",stu[i].name);
       for (j=0;j<3;j++)
          {printf("score %d:",j+1);
           scanf("%f",&stu[i].score[j]);
          }
      }
   average=0;
   max=0;
   maxi=0;
   for (i=0;i<N;i++)
      {sum=0;
       for (j=0;j<3;j++)
          sum+=stu[i].score[j];
       stu[i].avr=sum/3.0;
       average+=stu[i].avr;
       if (sum>max)
        {max=sum;
         maxi=i;
        }
      }
   average/=N;
   printf("    NO.        name    score1    score2    score3        average\n");
   for (i=0;i<N;i++)
      {printf("%5s%10s",stu[i].num,stu[i].name);
       for (j=0;j<3;j++)
          printf("%9.2f",stu[i].score[j]);
       printf("       %8.2f\n",stu[i].avr);
      }
      printf("average=%5.2f\n",average);
      printf("The highest score is : student %s,%s\n",stu[maxi].num,stu[maxi].name);
```

```
    printf("his scores are:%6.2f,%6.2f,%6.2f,average:%5.2f.\n",
stu[maxi].score[0],stu[maxi].score[1],stu[maxi].score[2],stu[maxi].avr);
  }
```

49 写一个函数，将一个字符串中的元音字母复制到另一个字符串，然后输出。

```
#include <stdio.h>
void copy_y(char a[], char b[]);

int main()
{
   char str1[20], str2[20];
   printf("请输入字符串： ");
   gets(str1);
   copy_y(str1, str2);

   return 0;
}

void copy_y(char a[], char b[])
{
     int i=0, j=0;
     while(a[i]!='\0')
     {
        if(a[i]=='a'|| a[i]=='e'|| a[i]=='i'|| a[i]=='o'|| a[i]=='u'
        ||a[i]=='A'|| a[i]=='E'|| a[i]=='I'|| a[i]=='O'|| a[i]=='U')
        {
          b[j] = a[i];
          j++;
        }
        b[j]='\0';
        i++;
     }

     if(j==0)
        printf("该字符串没有元音字母！ ");
     else
        printf("该字符串元音字母为： %s \n", b);
}
```

或者

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

```
void main()
{
    void copy(char s[3],char m[2]);
    char str[80],c[80];
    printf("input str:\n" );
    gets(str);
    copy(str, c);
    printf("the vovel letters are:%s\n",c);
}
void copy(char s[3],char m[2])
{
     int i,j;
     for(i=0,j=0;s[i]!='\0';i++)

if(s[i]=='a'||s[i]=='A'||s[i]=='o'||s[i]=='O'||s[i]=='u'||s[i]=='U'||s[i]=='i'||s[i]=='I'||s[i]=='e'||
s[i]=='E')
        {
          m[j]=s[i];
          j++;
        }
        m[j]='\0';
}
```

50 编一程序，输入月份号，输出该月的英文月名。例如，输入 3，则输出 March，要求用指针数组处理。

方法一：数组法

```
#include <stdio.h>
int main()
{
   int month;
   char month_name[][14] = {"illegal month","January", "February", "March",
"April",
   "May", "June", "July", "August", "September", "October", "November",
"December"};

   printf("请输入月份号：");
   scanf("%d", &month);

   if(month>=1 && month<=12)
      printf("月份对应英文为：%s", month_name[month]);
   else
    printf("输入错误，请检查！");

   return 0;
```

```
}
```
方法二：指针法
```
#include <stdio.h>
int main()
{char * month_name[13]={"illegal month","January","February","March","April",
    "May","June","july","August","September","October",
"November","December"};
int n;
printf("input month:\n");
scanf("%d",&n);
if ((n<=12) && (n>=1))
    printf("It is %s.\n",*(month_name+n));
else
   printf("It is wrong.\n");

   return 0;
}
```

51 编写完整程序：利用 2 个函数对输入的两个分数进行加、减、乘、除四则运算和输出用分数表示的结果。（注：输入格式为：%ld/%ld%c%ld/%ld，输出格式为%ld/%ld），例如：输 入 1/4+1/3，输出：7/12 （10 分）

```
#include <stdio.h>
long gcd(long m, long n);
void count(long a, char s1, long b, char s, long c, char s2, long d);

int main()
{
     long a, b, c, d;

     char s, s1, s2;
     printf("请输入计算式：\n");
      while((scanf("%ld %c %ld %c %ld %c %ld", &a, &s1, &b, &s, &c, &s2,
&d))!=EOF)
        count(a, s1, b, s, c, s2, d);

     return 0;
}

long gcd(long m, long n)
{
   long temp;
   if(m<n)
   {
     temp = m;
```

```
    m = n;
    n = temp;
  }
  while(n!=0)
   {
    temp = m%n;
    m = n;
    n = temp;
   }
  //最大公约数 m

 return m;
}

void count(long a, char s1, long b, char s, long c, char s2, long d)
{
    long lcm, temp, temp1;
    if(s1!='/' || s2!='/')
        printf("print error!");
    else
     switch(s)
     {
          case '+':
                    lcm = b*d/gcd(b, d);
                     a = a*(lcm/b);
                     c = c*(lcm/d);
                     temp = a+c;
                     temp1 = temp;
                     //计算机结果：temp/lcm;
                     temp = temp/gcd(lcm, temp);
                     lcm = lcm/gcd(lcm, temp1);
                     printf("=%ld / %ld\n", temp, lcm);
                     break;

          case '-':

                    lcm = b*d/gcd(b, d);
                     a = a*(lcm/b);
                     c = c*(lcm/d);
                     temp = a-c;
                     temp1 = temp;
                     //计算机结果：temp/lcm;
                     temp = temp/gcd(lcm, temp);
                     lcm = lcm/gcd(lcm, temp1);
```

```
                printf("=%ld / %ld\n", temp, lcm);
                break;

        case '*':
                a = a*c;
                b = b*d;
                //结果为  a/b
                temp = a;
                a = a/gcd(temp, b);
                b = b/gcd(temp, b);
                printf("=%ld / %ld\n", a, b);
                break;

        case '/':
        //a/b * d/c

                a = a*d;
                b = b*c;
                temp = a;
                a = a/gcd(temp, b);
                b = b/gcd(temp, b);
                printf("=%ld / %ld\n", a, b);
                break;


        default: printf("print error!\n");
    }

}
```

52 编写函数，将单链表进行逆序，即表头变表尾，表尾变表头（15 分）其中：
节点定义为：
struct node{int num, struct node *next };
函数原型为：void turn（struct node *head）

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

```
{
   int num;
   struct LNode *next;
};
void turn(LNode *head);

int main()
{
      int i;
      LNode *head;
     LNode *p, *q;
      head = (LNode *)malloc(sizeof(LNode));
      head->next=NULL;
      for(i=0; i<5; i++)
      {
          p = (LNode *)malloc(sizeof(LNode));
          p->num=i;
          p->next=head->next;
          head->next=p;
      }
     p = head;
     printf("逆转前：");
     while(p->next)
     {
        printf("%d ", p->next->num);
        p = p->next;
      }
      printf("\n");
     printf("逆转后：");
     turn(head);
     return 0;
}
void turn(LNode *head)
{
     //head 为头节点
     LNode *p = head->next, *q;
     head->next = NULL;
     while(p)
     {
          q = p->next;
          p->next = head->next;
          head->next = p;
          p = q;
     }
```

```
    //输出测试
    p = head;
    while(p->next)
    {
       printf("%d ", p->next->num);
       p = p->next;
     }

}
```

53 编写完整程序：接收从键盘输入的仅由数字字符构成的字符串（假设字符串的最大长度为 50），统计并输出每个数字（0~9）的重复次数。（8 分）

```
#include <stdio.h>
int main()
{
     char str[50];
     int a[10]={0}, i, temp;
     printf("请输入字符串： ");
     gets(str);
     while(str[i]!='\0')
     {
          if(str[i]>='0' && str[i]<='9')
          {
             temp = str[i]-'0';
            a[temp]++;
         }
          i++;
     }
     for(i=0; i<10; i++)
         printf("数字%d 的出现的次数为：%d\n", i, a[i]);
     return 0;
}
```

54 编写完整程序，采用结构数组和指向结构的指针，接收输入的 100 个学生信息（包括学号和 C 语言课程期末总成绩），输出最高、最低成绩和分别对应的学号（可能有多个同学都是最高分，可能有多个同学都是最低分）。（12 分）

方法 1：

```
#include <stdio.h>
#define N 6
typedef struct{
    char num[10];
```

```
    int cscore;
}*sPoint, stu;
void order(sPoint p, int *max, int *min);
void print_max_min(sPoint p, int max, int min);
int main()
{
    int i, max, min;
    stu s[N];
    sPoint p = s;
    printf("请输入%d 个学生信息：\n", N);
    printf("sno    cscore\n");
    for(i=0; i<N; i++)
    scanf("%s %d", (p+i)->num, &(p+i)->cscore);
    order(p, &max, &min);
    print_max_min(p, max, min);

    return 0;
}
void order(sPoint p, int *max, int *min)
{
    int i;
    *max = p->cscore;
    *min = p->cscore;
    for(i=0; i<N; i++)
    {
        if((p+i)->cscore>*max)
            *max = (p+i)->cscore;
        if((p+i)->cscore<*min)
            *min = (p+i)->cscore;
    }
}

void print_max_min(sPoint p, int max, int min)
{
  int i;
  printf("最高分成绩以及学生为：\n");
  for(i=0; i<N; i++)
  {
    if((p+i)->cscore==max)
      printf("学号：%s c 语言成绩：%d\n", (p+i)->num, (p+i)->cscore);


  }
```

```
    printf("最低分成绩以及学生为：\n");
    for(i=0; i<N; i++)
    {
      if((p+i)->cscore==min)
         printf("学号：%s c 语言成绩：%d\n", (p+i)->num, (p+i)->cscore);

    }

}
```

方法 2

```
#include <stdio.h>
#define N 6
struct stu{
      char num[10];
      int cscore;
};

int main()
{
     int i, max=0, min=100;
     stu s[N];
     printf("请输入%d 个学生信息：\n", N);
     printf("sno cscore\n");
     for(i=0; i<N; i++)
     {
        scanf("%s %d", s[i].num, &s[i].cscore);
       if(s[i].cscore>max)
            max = s[i].cscore;
       if(s[i].cscore<min)
            min = s[i].cscore;
     }
     printf("最高分学生以及其成绩为：\n");
     for(i=0; i<N; i++)
     {
       if(s[i].cscore==max)
          printf("学号：%s c 语言成绩：%d\n", s[i].num, s[i].cscore);
     }

     printf("最低分学生以及其成绩为：\n");
     for(i=0; i<N; i++)
     {
       if(s[i].cscore==min)
          printf("学号：%s c 语言成绩：%d\n", s[i].num, s[i].cscore);
```

```
    }

    return 0;
}
```

55 编写一个函数，使之能完成以下功能：把一个字符串逆序排列。(10 分)（保研题）

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

void creverse(char s[]);
int main()
{
    char str[100];
    int len;
    printf("请输入一个字符串：\n");
    gets(str);

    printf("该字符串的逆序为：\n");
    creverse(str);

    return 0;
}
void creverse(char s[])
{
   int i, len;
   char temp;
   len =strlen(s);
   for(i=0; i<len/2; i++)
   {
      temp = s[len-1-i];
      s[len-1-i] = s[i];
      s[i] = temp;
   }
   printf("%s", s);
}
```

备注：你电 14 年要求不允许申请新的数组空间（无所谓啦）

```
void reverse(char *str)
{
     int i=0, j=0;
     char c;
     while(str[j])j++;
```

```
    j--;
    while(i<j)
    {
        c=str[i];
        str[i]=str[j];
        str[j]=c;
        i++;j--;
    }
}
```

56 编写一个函数，使之能完成以下功能：利用递归方法找出一个数组中的最大值和最小值，要求递归调用函数的格式如下：MinMaxValue(arr,n,&max,&min)，其中 arr 是给定的数组，n 是数组的个数，max、min 分别是最大值和最小值。(15 分)（保研题）

方法 1：

```
#include <stdio.h>
void MinMaxValue(int arr[],int n,int *max, int *min);
#define N 10
int main()
{
    int a[10], i, max, min;
    printf("请输入%d 个整数：", N);
    for(i=0; i<N; i++)
       scanf("%d", &a[i]);
    max = min = a[0];
    MinMaxValue(a, N-1, &max, &min);

    printf("该数组最大值为：%d，最小值为：%d", max, min);
    return 0;
}
//注意 n 为数组下标，赋值时应把长度-1
void MinMaxValue(int arr[],int n,int *max, int *min)
{
  if(n>=0)
  {
      if(arr[n]>*max)
        *max = arr[n];
      if(arr[n]<*min)
        *min = arr[n];
      MinMaxValue(arr, n-1, max, min);
  }
}
```

方法 2：

```
#include<stdio.h>
#define N 10
void MinMaxValue(int arr[],int n,int *max,int *min)
{
    if(n==0) return;
    if(arr[n-1]>*max) *max=arr[n-1];
    if(arr[n-1]<*min) *min=arr[n-1];
    MinMaxValue(arr,n-1,max,min);
}
int main()
{
    int i, a[10], max, min;
    printf("请输入%d 个整数：", N);
     for(i=0; i<N; i++)
       scanf("%d", &a[i]);
     max=a[0], min=a[0];

    MinMaxValue(a,sizeof(a)/sizeof(int),&max,&min);
    printf("最大值为：%d，最小值为：%d\n",max,min);

    return 0;
}
```

或者：

```
void MinMaxValue(int arr[], int n, int *max, int *min)
{
    if(n==1)
    {
        *max = arr[0];
        *min = arr[0];
    }
    else
    {
        int _max=arr[0], _min=arr[0];
        MinMaxValue(arr+1, n-1, max, min);
        if(*max<_max)*max=_max;
        if(*min>_min)*min=_min;
    }
}
```

57 编写一个函数，使之能完成以下功能：把 file1.doc 的内容全部复制到 file2.doc 中，file1.doc 中全部是字符(含空格)，要求复制时，在 file2.doc 中的每一行都要加上行号，例如：行号*(其中“*”表示具体的数字)。最后该函数返回 file1.doc 中的字符个数(不包括空格)。(10 分)（保研题）

方法一：本代码仅支持 txt 文本文件有效，doc 文件会出问题

```
#include "stdio.h"
/*
"r"：只能从文件中读数据，该文件必须先存在，否则打开失败
"w"：只能向文件写数据，若指定的文件不存在则创建它，如果存在则先删除它再重建一个新文件
"a"：向文件增加新数据(不删除原有数据)，若文件不存在则打开失败，打开时位置指针移到文件末尾
"r+"：可读/写数据，该文件必须先存在，否则打开失败
"w+"：可读/写数据，用该模式打开新建一个文件，先向该文件写数据，然后可读取该文件中的数据
"a+"：可读/写数据，原来的文件不被删去，位置指针移到文件末尾
*/
int exam()
{
    char ch;
    int count=0,row=0;
    FILE *fp1,*fp2;
    fp1=fopen("file1.txt","r+");
    //利用 feof()函数检查文件是否读取完毕
    while(!feof(fp1))
    {
        ch=getc(fp1);
        fp2=fopen("file2.txt","a+");
        if(ch!=' ')
            count++;
        if(ch=='\n')
        {
/*它们与 printf 和 scanf 函数相仿，都是格式化读写函数。不同的是：fprintf 和 fscanf 函数
的读写对象不是终端(标准输入输出)，而是磁盘文件。printf 函数是将内容输出到终端(屏幕)，
因此，fprintf 就是将内容输出到磁盘文件了。    */
            row++;
            fprintf(fp2,"行号:%d",row);
            fprintf(fp2,"\n");
        }
        else
        {
            putc(ch,fp2);
        }
        fclose(fp2);
    }
```

```
    fclose(fp1);
    return count;
}

int main()
{
    int sum=exam();
    printf("Sum=%d\n",sum);
    return 0;
}
```

58 编写一个完整的程序，使之能完成以下功能：从键盘中输入若干个整数，用链表储存这些输入的数，并要求存储的顺序与输入的顺序相反。(10 分)（保研题）

```
#include <stdio.h>
#include <stdlib.h>
#define N 10

typedef struct LNode{
    int num;
    struct LNode *next;
}LinkList;
void creat_L(LinkList * &L, int a[], int n);
void print_L(LinkList * &L);

int main()
{
   int i, a[N];
   LinkList *L;
   printf("请输入%d 个整数：", N);
   for(i=0; i<N; i++)
       scanf("%d", &a[i]);
   creat_L(L, a, N);
   printf("创建后的链表如下：");
   print_L(L);

   return 0;
}

void creat_L(LinkList * &L, int a[], int n)
{
```

```
    int i;
    LinkList   *p;
    L = (LinkList *)malloc(sizeof(LinkList));
    L->next = NULL;
    for(i=0; i<n; i++)
    {
      p = (LinkList *)malloc(sizeof(LinkList));
      p->num = a[i];
      p->next = L->next;
      L->next = p;
    }

}
void print_L(LinkList * &L)
{
     LinkList *p = L->next;
     while(p != NULL)
     {
          printf("%d ", p->num);
          p = p->next;
     }
}
```

59 编写一个完整的程序，使之能完成以下功能：一段名为 file.c 的程序，该程序中含有括号，现要检查程序中的括号是否配对，提示：利用堆栈实现。(15 分)（保研题）

```
#include <stdio.h>
#include <string.h>
#define Stack char
#define Max 999
int judge(char p[])
{
     int len=strlen(p);
     int top=-1,i;
     Stack s[Max];
     for(i=0; i<len; i++)
     {
          switch(p[i])
          {
```

```
                case '(':
                case '[':
                case '{':
                        s[++top]=p[i];
                        break;
                case ')':
                        if(s[top]=='(')
                                top--;
                        else return 0;
                        break;
                case ']':
                        if(s[top]=='[')
                                top--;
                        else return 0;
                        break;
                case '}':
                        if(s[top]=='{')
                                top--;
                        else return 0;
                        break;
                }
        }
        if(top==-1)
                return 1;
        else
                return 0;
}

int main()
{
        int i=0;
        char p[Max];
        FILE * fin;
        if(!(fin=fopen("file.txt","r")))
                printf("failed");
        while(!feof(fin))
        {
                //fscanf(fin,"%s",p);
                /* 我把你电的代码修正了一下，如果使用上述函数读取，
                只能读取一行，如果文本有换行则会出错，而 fgetc 函数全盘读取，没得
问题*/
                p[i] = fgetc(fin);
                i++;
```

```
    }
    if(judge(p))
    printf("括号匹配！\n");
    else
    printf("括号不匹配！\n");
    fclose(fin);

    return 0;
}
```

60 将输入的一个数质因分解，如 Input 90，Print:90=2*3*3*5.（保研题附加题）

方法一：

```
#include <stdio.h>
int panduan(int n);
int main()
{
    int i, in, j, a[100];
    printf("请输入一个整数：\n");
    while(scanf("%d",&in)&&in>-1)
    {
     if(in==0 || in==1)
        printf("该数没有质因数！");
     else
         {
          j=0;
          printf("%d=", in);
          while(in>3)
          {
              for(i=2; i<=in; i++)
             {
               if(panduan(i)&&in%i==0)
                  break;
             }
             in = in/i;
             a[j++] = i;
          }
          if(in>1)
              a[j++]=in;
           for(i=0; i<j; i++)
           {
              printf("%d", a[i]);
              if(i<j-1)
              printf("*");
```

```
            }

          }
        printf("\n");
    }

    return 0;
}

int panduan(int n)
{
 int i,temp;
 temp=1;
 for(i=2;i*i<=n&&temp==1;i++)
       if(n%i==0)
            temp=0;
     return (temp);
}
```

方法 2

```
# include <stdio.h>

int main()
{
     int i,j,k,val;
    printf("请输入一个正整数：");
    scanf("%d",&val);

    printf("将该正整数分解质因数输出为：  ");
    printf("%d=",val);
    //以下为算法
    for(i=2 ;i<=val;i++)
    {
        while(val!=i)
        {
                if(val%i == 0)
            {
                printf("%d*",i);
                val=val/i;
            }
            else
                break ;
        }
    }
```

```
    printf("%d",val);
    printf("\n");
    return 0;
}
```

61 编程求 $sum = 1 - x + \frac{x^2}{2!} - \frac{x3}{3!} .... + \frac{x^n}{n!}$， x,n 键盘输入。（保研题附加题）

方法一：

```
#include <stdio.h>
float factorial(float n);
float pow_n(float x, float n);

int main()
{
    float x, i;
    int n;
    float sum=0;
    printf("x=");
    scanf("%f", &x);
    printf("n=");
    scanf("%d", &n);
    for(i=0; i<=n; i++)
    {
    sum += (pow_n(-1, i))*(pow_n(x, i))/(factorial(i));
    }
     printf("sum=%f", sum);
    return 0;
}

float factorial(float n)
{
    float pro=1;
    if(n==0 || n==1)
       return 1;
    while(n!=1)
    {
       pro *= n;
       n--;
    }
    return pro;
}
float pow_n(float x, float n)
{
    float pow=1;
```

```
    int i;
    if(n==0)
       return 1;
    for(i=0; i<n; i++)
        pow *= x;
    return pow;
}
```

方法 2

```
# include <stdio.h>
int main()
{
    float sum=1, sum1=1, sum2=1, sum3, x;

    int i, n, flag=-1;

    printf("请输入 x 的值：\n");
    scanf("%f", &x);

    printf("请输入 n 的值：\n");
    scanf("%d", &n);

    for(i=1; i<=n; i++)
    {
        sum1 = sum1*x;
        sum2 = sum2*i;
        sum3 = sum1/sum2 * flag;
        flag = -flag;
        sum = sum + sum3;
    }
    printf("Sum=%f\n", sum );
    return 0;
}
```

方法三：大佬做法压轴

```
#include<stdio.h>
int main()
{
    float sum=1, x, item=1;
    int n, i;
    scanf("%f %d",&x, &n);
    for(i=1; i<=n; i++)
    {
        item *= -1*x/i;
        sum += item;
```

```
    }
    printf("sum=%f\n", sum);
    return 0;
}
```

62 删除输入的字符串中的大小写字母和数字，并统计有重复的字符及其重复次数

//他说的统计重复字符说的不清楚，到底是什么字符，难道是任意重复字符吗？？？那怎么统计

//15 年也有个类似题：删除字符串中的英文和数字并将重复字符减为一个

```
#include <stdio.h>
#include <string.h>
void delete_c(char s[], int i, int n);
int main()
{
    int i, len;
    char str[100];
    printf("请输入字符串： ");
    gets(str);
    len = strlen(str);
    while(str[i]!='\0')
    {
            if((str[i]>='a'&&  str[i]<='z')||(str[i]>='A'&&  str[i]<='Z')||(str[i]>='0'&&
str[i]<='9'))
          delete_c(str, i, len);
        else
          i++;
    }

    printf("删除后字符串为： %s", str);
    return 0;
}
void delete_c(char s[], int i, int n)
{
   int j;
   for(j=i; j<n; j++)
      s[j] = s[j+1];
}
```

63 输入整形数据，按输入的逆序建立单链表（这个题和前面题一样，所以我把题目改成顺序的，下面代码是顺序）

```
#include <stdio.h>
#include <stdlib.h>
#define N 10
typedef struct LNode{
      int num;
      struct LNode *next;
}LinkList;
void creat_L(LinkList * &L, int a[], int n);
void print_L(LinkList * &L);

int main()
{
     int i, a[N];
     LinkList *L;
     printf("请输入%d 个整数：", N);
     for(i=0; i<N; i++)
        scanf("%d", &a[i]);
     creat_L(L, a, N);
     printf("输出创建后的数组为：");
     print_L(L);
      return 0;
}

void creat_L(LinkList * &L, int a[], int n)
{
      int i;
      LinkList *s, *r;
      L = (LinkList *)malloc(sizeof(LinkList));
      r = L;
      for(i=0; i<n; i++)
      {
            s = (LinkList *)malloc(sizeof(LinkList));
            s->num = a[i];
            r->next = s;
            r = s;
      }
      r->next = NULL;

}

void print_L(LinkList * &L)
{
      LinkList *p = L->next;
      while(p != NULL)
```

```
    {
        printf("%d ", p->num);
        p = p->next;
    }
}
```

64 身份证长度是 18，其中第 7 至 14 个数字是生日，编写完整程序判断 2 个身份证号的出生日期先后。

```
#include<stdio.h>
int isbothsame(char str1[19], char str2[19])
{
    int low=6, hight=13;
    while(low<hight && str1[low]==str2[low])low++;
    return (str1[low]-str2[low]);
}
int main()
{
    char str1[19], str2[19];
    printf("请输入第一个人身份证：\n");
    scanf("%s", str1);
    printf("请输入第二个人身份证：\n");
    scanf("%s", str2);
    printf("比较结果为：");
    if(isbothsame(str1, str2)<0)
        printf("第一个人先出生！");
    else if(isbothsame(str1, str2)>0)
        printf("第二个人先出生！");
     else
        printf("两个人同时出生！");
    return 0;
}
```

65 编写完整程序，一个链表，找出其中数据项最大的结点，然后将其移动到链表尾部（结点 node 由整型 data 和节点指针 next 构成），不允许申请新的结点。

方法一：不断链

```
void movemax(node *L)
{
    node *p=L->next, *pre=L, *max=p;
    while(p)
    {
        if(p->data>max->data)max=p;
        pre=p;
        p=p->next;
    }
```

```
    int temp = pre->data;
    pre->data = max->data;
    max->data = temp;
}
```

方法二：断链

```
void maxInsert(Lnode *L)
{
  Lnode *p,*q,*pre;//pre 记录最大值节点的前驱指针
  int max;
  max=L->data;
  q=L;//工作指针
  p=L;//记录最大值结点
  while（p->next!=null)
  {
    if(p->next ->data >max)
    {
      max=p->next ->data ;
      q=p->next ;
      pre=p;
    }
    p=p->next ;
  }

  pre->next =q->next ;//删除 q 结点即最大值结点
  p->next =q;
  q->next=null;
}
```

66 编写一个函数，把整数序列分成两个部分，使得左边部分都不大于右边部分，不需要排序。 （ 考察的是快速排序的部分）

```
int partion(int arr[], int n)
{
    int pos=0;
    int i=0, j=n-1;
    int temp;

    while(i!=j)
    {
        while(arr[j]>=arr[pos] && i<j)j--;
        while(arr[i]<=arr[pos] && i<j)i++;
        if(i<j)
        {
            temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
```

```
            }
        }

        temp = arr[i];
        arr[i] = arr[pos];
        arr[pos] = temp;

        return 0;
}
```

67 有两个整数数组 A 和 B，它们分别有 m、n 个整数。并且都是按非递减序列，现将 B 数组插入 A 数组中，使得 A 数组中各元素不大于 B 数组中各元素，且还是非递减序列。（如果去掉“使得 A 数组中各元素不大于 B 数组中各元素”，我还是可以愉快的写答案的）

68 两个递增有序整数数列链表 La 和 Lb，将他们合并后，变成一个新的链表，要求该链表递减排序。（结点 node 由整型 data 和节点指针 next 构成）

```
typedef struct node
{
    int data;
    struct node *next;
}node;

node *Union(node    *La, node    *Lb)
{
    node *p=La->next;
    node *q=Lb->next;
    node *temp;
    La->next=0;

    while(p&&q)
    {
        if(p->data <= q->data)
        {
            temp = p->next;
            p->next = La->next;
            La->next = p;
            p=temp;
        }
        else
```

```
            {
                temp = q->next;
                q->next = La->next;
                La->next = q;
                q=temp;
            }
        }

        if(q)p=q;
        while(p)
        {
            temp = p->next;
            p->next = La->next;
            La->next = p;
            p=temp;
        }
        free(Lb);
        return La;
}
```

69 编写一个函数，删除链表中的最小值。（结点 node 由整型 data 和节点指针 next 构成）

```
void delmin(node *L)
{
    node *p=L->next, *pre=L;
    node *min=p, *minpre=L;
    while(p)
    {
        if(p->data < min->data)
        {
            min = p;
            minpre = pre;
        }

        pre = p;
        p = p->next;
    }

    minpre->next = min->next;
    free(min);
}
```

70 编写函数判断小括号是否匹配。

注：这道题由于限定了只有小括号，故不需要栈来实现

```
int ismarry(const char *str)
{
```

```
    int top=-1;
    char c;

    while(*str)
    {
        if(*str == '(')++top;
        if(*str == ')')
        {
            if(top == -1)return -1;
            top--;
        }
        str++;
    }
    if(top == -1)return 0;
    return -1;
}
```

71 对多个字符串进行字典排序

方法一：

```
void Sort(char *parr[],int n)
{
    int i, j;
    char *str1, *str2;
    for(i=0; i<n-1; i++)
    {
        for(j=i+1; j<=n-1; j++)
        {
            str1=parr[i];
            str2=parr[j];
            while(*str1 && *str2 && *str1==*str2)str1++,str2++;
            if(*str1-*str2>0)
            {
                char *temp = parr[i];
                parr[i] = parr[j];
                parr[j] = temp;
            }
        }
    }
}
```

方法二：

```
void sort(char str[][100])
{
int i, j;
char temp[100];
//冒泡排序
```

```
for(i = 0; i < 9; i++)
{
   for(j = i+1; j < 10; j++)
   {
        if(strcmp(str[i], str[j]) == 1)//strcmp 比较字符串 str1 和 str2 是否相同。如果相同则返回 0；
                                                    //如果不同，在不同的字符串处如果 str1 的字符大于 str2 的字符，则返回 1，否则返回-1
        {
            strcpy(temp, str[i]);
            strcpy(str[i], str[j]);
            strcpy(str[j], temp);
        }
   }
}
}
```

72 有两字符数组 s 和 t，求 t 在 s 中出现第一次的开始位置，如果没有则输出“No”，有则输出开始位置。

```
int start(char s[],char t[])
{
    int s_length = strlen(s);
    int i;
    char *str1, *str2;
    for(i=0; i<s_length; i++)
    {
        str1 = s+i;
        str2 = t;
        while(*str1 && *str2 && *str1==*str2)str1++,str2++;
        if(*str1-*str2 == *str1)return i;//str2 到头了
    }
    printf("NO!\n");
    return -1;
}
```

73 编写一个函数，从字符串中寻找整个连续的数字字符，将其转化成整数并保存在 arr 整型数组里，溢出以-1 作为标志。字符串以‘#’结束（如"uestc2015jsj123#"）

```
void int_find(int arr[], char *pc)
int str2int(char *str, int len)
{
    int i, num=0;
    for(i=len-1; i>=0; i--)
        num += (str[len-i-1]-48)*pow(10.0, i);
    return num;
```

```
}

void int_find(int arr[], char *pc)
{
    char *p=pc, *q;
    int i=0;
    while(*p!='#')
    {
        if('0'<=*p && *p <= '9')
        {
            q=p+1;
            while('0'<=*q && *q <= '9')q++;
            arr[i++] = str2int(p, q-p);
            p=q;
        }
        else
        {
            p++;
        }
    }
    arr[i]=-1;
}
```

74 随机输入若干整数和若干浮点数（顺序也是随机的，最多 100 个），要求编写完整的程序，将整数按从大到小排列，浮点数按从小到大排列（浮点数排序可省略），并输出。如：输入 10 12.3 12 5 52.1 3.65 88.6 输出：12 10 5 3.65 12.3 52.1 88.6

这道题我采用的是一种简单的判断输入的数是整数还是浮点数的方法，但是该算法有个缺陷就是会将 10.0 认作整数 10。但是考虑在考试中时间有限无法使用其它耗时的方法（比如读取成字符串然后判断有无字符'.'从而判断是否为浮点数）。并且，在原题中给出的数字并未出现 10.0 这样的情况，故综上，我选择这样的方法（具体见代码）。（下面还有一个差不多的题，你们可以看看）

```
void sort_int (int num[], int n)
{
    int i, j;
    for(i=0; i<n-1; i++)
    {
        for(j=i+1; j<=n-1; j++)
        {
            if(num[j]<num[i])
            {
                int temp = num[i];
                num[i] = num[j];
                num[j] = temp;
            }
```

```
            }
        }
}

void main()
{
    float temp;
    int num_int[100], count_int=0, count_foat=0;
    float num_float[100];
    int i, n;

    scanf("%d", &n);
    for(i=0; i<n; i++)
    {
        scanf("%f", &temp);
        if(temp == (int)temp)
            num_int[count_int++]=(int)temp;
        else
            num_float[count_foat++]=temp;
    }

    sort_int(num_int, count_int);
    sort_float(num_float, count_foat);
    for(i=0; i<count_int; i++)
        printf("%d ", num_int[i]);
    for(i=0; i<count_foat; i++)
        printf("%f", num_float[i]);
}
```

75 编写完整的程序，构造整数集合(其实就是一个整数链表)，并实现对该集合操作的若干功能：加入一个新元素，判断某元素是否在集合内，输出两个集合并集，输出两个集合交集，删除集合中某一元素。

struct set{int numb; struct set *next;}

注意：链表里是没有重复数字的，因为题目中说了是集合，集合是不会有重复元素出现的。此外，本题有时间空间复杂度更优的算法，但是在考试时写出这样的算法不太现实，故我这里提供的还是普通版的。

```
//加入一个新元素
void add(struct set *L, int num)
{
    struct set *p=(struct set *)malloc(sizeof(struct set));
    p->numb=num;
    p->next = L->next;
    L->next = p;
```

```
}

//判断某元素是否在集合内
int isexist(struct set *L, int num)
{
    struct set *p=L->next;
    while(p)
    {
        if(p->numb == num)return 1;
        p=p->next;
    }
    return 0;
}

//输出两个集合并集
void intersection(struct set *a, struct set *b)
{
    struct set *p=a->next;
    int flag=0;
    while(p)
    {
        printf("%d ", p->numb);
        p=p->next;
    }
    struct set *q=b->next;
    while(q)
    {
        p=a->next;
        flag=0;
        while(p)
        {
            if(q->numb == p->numb)
            {
                flag=1;
                break;
            }
            p=p->next;
        }
        if(!flag)printf("%d ", q->numb);
        q=q->next;
    }
}

//输出两个集合交集
```

```
void Union(struct set *a, struct set *b)
{
    struct set *p=a->next, *q=b->next;
    while(p)
    {
        q=b->next;
        while(q)
        {
            if(p->numb == q->numb)
            {
                printf("%d ", q->numb);
                break;
            }
            q=q->next;
        }
        p=p->next;
    }
}

//删除集合中某一元素
void del(struct set *a, int num)
{
    struct set *p=a->next, *pre=a;
    while(p)
    {
        if(p->numb == num)
        {
            pre->next=p->next;
            free(p);
            break;
        }
        pre=p;
        p=p->next;
    }
}

void main()
{
    struct set *L = (struct set *)malloc(sizeof(struct set));
    struct set *S = (struct set *)malloc(sizeof(struct set));
    L->next=0;
    S->next=0;

    int temp;
```

```
    scanf("%d", &temp);
    while(temp!=-1)
    {
        add(L, temp);
        scanf("%d", &temp);
    }

    scanf("%d", &temp);
    while(temp!=-1)
    {
        add(S, temp);
        scanf("%d", &temp);
    }
    printf("\n 并集： ");
    intersection(L, S);
    printf("\n 交集： ");
    Union(L, S);
}
```

76 凯撒加密

这道题是一道加密解密问题，大致的意思就是通过主函数输入两个字符串，第一个字符串作为加密解密的目标字符串，第二个字符串通过函数转换为整型数据作为解密加密的 key，这道题也不难推导，以上是我在网络上找到的一个相似版本。凯撒加密(Caesarcipher)是一种简单的消息编码方式：它根据字母表将消息中的每个字母移动常量位 k。举个例子如果 k 等于 3，则在编码后的消息中，每个字母都会向前移动 3 位：a 会被替换为 d；b 会被替换成 e；依此类推。字母表末尾将回卷到字母表开头。于是，w 会被替换为 z，x 会被替换为 a。源代码（看懂就好，不必太深究）：

```
#include <stdio.h>
#include <stdlib.h>
int main () {
   char small_letter[26] = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l',
'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};
   char big_letter[26] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'};
   char text[1000], result[1000];

   int c, count = 0, k, p;
   char function;
   printf("Insert Text:");
   c = getchar();
   while (1) { // 读取字符串
       if (c == '\n') break;
```

```
        text[count] = c;
        printf("%c", text[count]);
        count++;
        c = getchar();
}

  printf("\n");
  printf("Encrypt or Decrypt? E or D :");
  scanf("%c", &function);
  if (function == 'E')
  {
     printf("Insert Key :" );
     scanf("%d", &k);
     for (int i = 0; i < count; i++)
     {
        if (text[i] >= 'A' && text[i] <= 'Z') {
           result[i] = big_letter[((text[i] - 'A') + k) % 26];
        }
//找出加密后字符在字符数组?的对应位置
        else if (text[i] >= 'a' && text[i] <= 'z') {
           result[i] = small_letter[((text[i] - 'a') + k) % 26];
         }
        else result[i] = text[i];
        printf("%c", result[i]);
     }
}
     else {
     printf("Insert Key :" );
     scanf("%d", &k);
     for (int i = 0; i < count; i++)
    {
        if (text[i] >= 'A' && text[i] <= 'Z') {
        p = ((text[i] - 'A') - k);
        while (p < 0)p += 26;
        result[i] = big_letter[p];
        }
//找出解密后字符在字符数组?的对应位置
//这?要注意不要让它超出范围（下表位置为负数）
        else if (text[i] >= 'a' && text[i] <= 'z') {
        p = ((text[i] - 'a') - k);
        while (p < 0)p += 26;
        result[i] = small_letter[p];
     }
        else result[i] = text[i];
```

```
        printf("%c", result[i]);
    }
        printf("\n");
    }
    return 0;
}
```

77 输入若干个整数（以 0 结束）如何逆序构建双向链表

```
#include <stdio.h>
#include <stdlib.h>
typedef struct DNode {
    int data;
    struct DNode *pre;
    struct DNode *next;
} DNode;

int main() {
    DNode *head, *s;
    int t;
    head = (DNode*)malloc(sizeof(DNode));
    head->pre = NULL;
    head->next = NULL;
    scanf("%d", &t);
    while (t != 0) {
        s = (DNode*)malloc(sizeof(DNode));
        s->data = t;
        s->next = NULL;
        s->pre = NULL;
        if (head->next == NULL) {
            head->next = s;
            s->pre = head;
        }
        else {
            s->next = head->next;
            head->next->pre = s;
            head->next = s;
            s->pre = head;
          }
        scanf("%d", &t);
}
        s = head->next;
        while (s) {
```

```
        printf("%d ", s->data);
        s = s->next;
    }
        return 0;
}
```

78 要求必须是 log2n 时间复杂度

思路：二分法

1、判断最小的数是否在第一个位置，若是则返回 nums[0];

2、否则，选取中间位置 mid 的数和 r 位置的数进行比较，若 nums[mid] < nums[r]，说明最小的数在 mid 之前，r = mid；反之，令 l = mid+1;

3、重复进行，直到 l >= r。

```
int l = 0;
int r = nums.length - 1;
if (nums[l] < nums[r]){
    return nums[l];
}
while(l < r){
    int mid = (l+r)/2;
    if(nums[mid] > nums[r]){
        l = mid + 1;
    }else{
        r = mid;
    }
}
```

这个问题主要分成如下三种情况

- $l < mid > r$
- $l > mid < r$
- $l < mid < r$

如果发生旋转的话，那么 nums[r]<nums[l]一定成立（非常容易证明），所以对于第一种情况最小值一定在右侧，所以我们更新 low=mid+1，对于第二种情况最小值一定在左侧或者就在 mid，这要怎么理解呢？如果最小值在右侧的话，说明 nums[mid]比最小值大，那么 nums[mid]一定是大于 nums[low]的（与题目条件不符），所以原结论成立，所以我们更新 high=mid。对于第三种情况，最小值一定是 nums[low]。
但是上述结论仅适用于 len(nums)>=3 的情况，对于 len(nums)<3 的情况，我们

要加以修正。

我查阅了 csdn,有几个博主讲的两个方法很好。

先来常规的：

原文链接：https://blog.csdn.net/SoYangA/article/details/81542081

1、利用在栈上创建一个跟原本一样的数组，然后按旋转后的数组给本身数组重新赋值

```
void rotate(int* nums, int numsSize, int k) {
    int temp[numsSize],p=0;
    for(int i=0;i<numsSize;i++)
        temp[i]=nums[i];
    for(int j=0;j<numsSize;j++){
        p=(j+k)%numsSize;
        nums[p]=temp[j];
    }
    return *nums;
}
```

2、在动态内存内创建数组然后利用 memcpy 函数进行拷贝（别忘了引头函数

```
#include <string.h>)
void rotate(int* nums, int numsSize, int k)
{
    k = k%numsSize;
    int* t = (int*)malloc(sizeof(int)*k);
    for (int i = 0; i < k; i++)
    {
        t[i] = nums[numsSize - k + i];
    }
    for (int i = numsSize - 1; i >= k; i--)
    {
        nums[i] = nums[i - k];
    }
    memcpy(nums, t, sizeof(int)*k);
    free(t);
}
```

3、利用交换函数，交换 3 次即可得到反转的函数

```
void reverse(int*left, int*right){
    assert(left&&right);
    while (left <= right){
        int temp = *left;
        *left = *right;
        *right = temp;
        left++;
```

```
        right--;
    }
}
void rotate(int* nums, int numsSize, int k) {
    k = k%numsSize;
    if (k == 0)
    {
        return;
    }
    reverse(&nums[0], &nums[numsSize - k - 1]);
    reverse(&nums[numsSize - k], &nums[numsSize - 1]);
    reverse(&nums[0], &nums[numsSize - 1]);
}
```

附下文连接:https://blog.csdn.net/qq_28584889/article/details/83655019
下文代码击败 99%

```
4、void rotate(int* nums, int numsSize, int k) {
        int temp1, temp2, index, count = 0;//count 为计数变量，记录移动成功的次数，temp1 和 temp2 是辅助存储变量，为交换元素值之用
        int len = numsSize;
        k %= len;//对 k 值的前期处理，处理原因是如果 k 值比 len 大，那么只有大于 len 的那部分才是有效移动
        if(k == 0)
            return;//如果 k 等于 0 或者 k 原本是等于 len 的，那么就相当于没有移动嘛，直接返回
        for (int i = 0; i <= k; i++) //移动的轮数最多 k 次，当然计数变量 count=len 的时候会跳出循环
        {
            if(count >= len)
                break;//对计数变量的控制，当所有位置全部移动完了就可以结束了
            index = i;//每轮移动的初始下标
            temp1 = nums[i];
            while((index + k)%len != i) //一个 while 循环移动一次
            {
                temp2 = nums[(index + k)%len];
                nums[(index + k)%len] = temp1;
                index = (index + k)%len;
                temp1 = temp2;
                count++;
            }
            nums[i] = temp1;
            count++;
```

```
        }
    }
```

79

简述算法思想，并编写完整程序。
输入 10 个数（包括整数和浮点数，但整数和浮点数的个数不确定），存放在一个数组中。

1. 按照从小到大的顺序输出整数，从大到小的顺序输出浮点数（可以省略浮点数的排序代码）。
   例如输入：`12 3 445.8 3.0 5.6 22 3.14 22.2 100.5 22.6`
   则输出：`3 12 22 455.8 100.5 22.6 22.2 5.6 3.14 3.0`
2. 按照输入顺序将浮点数存放到一个浮点数链表中。

要求：

1. 简述算法思想；
2. 编写完整程序；
3. 采用字符串数组，输入的 10 个数都当作字符串；
4. 字符串要转换为整数或浮点数；
5. 排序

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define EPS 1e-6
typedef struct floatLinkList{
    float num;
    struct floatLinkList *next;
}LinkList;
//整数排序
void sort_int(int num[], int m)
{
    int i, j;
    int temp;
    for(i=0; i<m-1; i++){
        for(j=i+1; i<=m-1; j++){
          if(num[j]<num[i])
          {
            temp = num[i];
            num[i] = num[j];
             num[j] = temp;
          }
       }

    }
```

```
}


//浮点排序
void sort_float(float num[], int m)
{
    int i, j;
    float temp;
    for(i=0; i<m-1; i++)
    {
        for(j=i+1; i<=m-1; j++)
        {
            if(num[j]-num[i]>EPS)
            {
              temp = num[i];
              num[i] = num[j];
              num[j] = temp;
            }

        }

    }
}

int main()
{
    LinkList *head = (LinkList *)malloc(sizeof(LinkList));
    LinkList *r = head;
    char num_c[100];
    int num_i[10];
    float num_f[10];
    int m=0, n=0, i;//m 为整形数组大小，n 为浮点型数组大小
    printf("请输入十个数：\n");
    for(i=0; i<10; i++)
    {
      scanf("%s", num_c);
//extern char *strstr(char *str1, const char *str2);
//返回值：若 str2 是 str1 的子串，则返回 str2 在 str1 的首次出现的地址；
//如果 str2 不是 str1 的子串，则返回 NULL

       if(strstr(num_c, ".")){
         LinkList *s = (LinkList*)malloc(sizeof(LinkList));
```

```
            s->num = atof(num_c);//字符串转浮点型存入链表
            r->next = s;
            r = s;
            num_f[n++] = atof(num_c);
        }else{
            num_i[m++] = atoi(num_c);
        }

    }
        r->next = NULL;
        sort_int(num_i, m);
        sort_float(num_f, n);

        for(i=0; i<m; i++)
            printf("%d", num_i[i]);

        for(i=0; i<n; i++)
            printf("%f", num_f[i]);

         printf("\n");

          r = head->next;
          while(r)
          {
              printf("%f\n", r->num);
              r = r->next;
          }

        return 0;
    }
```