

知识点 1 【++ --】

1、i++ i-- ++i --i

如果 i++ 或 ++i 独立的作为一条语句 没有区别

```
void test01()
{
    int i=3;
    //i++;
    ++i;
    printf("i=%d\n", i); //4
}
```

2、如果++在左边 ++i

先加减 后使用。

```
void test02()
{
    int i=3;
    int j=0;
    j=++i; //i=i+1; j=i;
    printf("i=%d, j=%d\n", i, j); //i=4 j=4
}
```

```
2. 10.9.42.114
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work/c/day02$ gcc 00_code.c
edu@edu:~/work/c/day02$ ./a.out
i=4, j=4
edu@edu:~/work/c/day02$
```

3、如果++在右边 i++

先试用 后加减

```
void test02()
{
    int i=3;
    int j=0;
    j=i++; //j=i; i=i+1;
    printf("i=%d, j=%d\n", i, j); //i=4 j=3
}
```

```
2. 10.9.42.114
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work/c/day02$ gcc 00_code.c
edu@edu:~/work/c/day02$ ./a.out
i=4, j=3
edu@edu:~/work/c/day02$
```

扩展：

```
18 void test03()
19 {
20     int a=10;
21     a=a++; //将a的值赋值给a 后面的自增语句失效 所以a=10
22     printf("a=%d\n", a); //10
23 }
```

知识点 2【数组的概述】

1、数组的概述

用一段连续的空间 存储相同类型的值的容器 叫做数组。（背）

数组的每一个元素 等价 普通变量。

```
int data1=10;  
int data2=20;  
int data3=30;  
.....  
int data5 = 50;
```

arr数组名		元素				
		10	20	30	40	50
0	1	2	3	4	数组的元素下标	
arr[0]	arr[1]	arr[2]	arr[3]	arr[4]		

2、数组的定义（步骤）

1、数组名和[]表示数组，[]里面必须给定元素的个数

```
arr[5]
```

2、数组每个元素为啥类型 就用该类型定义一个变量

```
int data;
```

3、从上往下整体替换

```
int arr[5];
```

案例 1：定义一个数组 arr 有 5 个元素 每个元素为 int *类型

```
int *arr[5];
```

案例 2：定义一个数组 arr 有 5 个元素 每个元素又为数组 该数组有 5 个元素 每个元素为

```
int
```

```
int arr[5][5];
```

案例 3：定义一个数组 arr 有 5 个元素 每个元素为 struct stu 类型

```
struct stu arr[5];
```

案例 4：定义一个数组 arr 有 5 个元素 每个元素为函数指针 该函数为 int fun(int,int)

```
int (*arr[5])(int,int);//函数指针数组
```

知识点 3 【一维数值数组】

1、一维数值数组的定义

```
#include<stdio.h>

void test01()

{

//一维数值数组 局部数组不初始化 元素内容不确定

int arr[5];

//定义数组的时候 []里面不能有变量比如：int n=5; int arr[n];

//数组的下标范围：0~4 （背）

//数组的元素范围：arr[0]~arr[4] （背）


//数组名都是符号常量 不能被赋值（记）

//arr=100;//error


//sizeof 测量类型的大小

//数组名作为类型 需要和 sizeof 结合：sizeof(arr) == 数组的总大小 == 数组元素的个数

*每个元素的大小

//数组名作为类型代表的是数组的总大小（记）

printf("%ld\n",sizeof(arr));//20
```

```
//数组元素的个数 = sizeof(arr)/sizeof(arr[0]) (记)

int n = sizeof(arr)/sizeof(arr[0]);

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


int i=0;

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

{

    printf("%d ", arr[i]);

}

printf("\n");

}
```

2、一维数值数组的初始化

1、全部初始化：给数组的每个元素都初始化

```
//全部初始化

//int arr[5]={10,20,30,40,50};

//如果全部初始化 可以省略元素个数

//数组元素的个数 由初始化个数决定

//int arr[]={10,20,30,40,50};
```

2、部分初始化

```
//部分初始化: 未被初始化的部分自动补 0

int arr[5]={10,20,30}; //10 20 30 0 0

int arr[5]={0}; //推荐: 将第 0 个元素初始化为 0 其他未被初始化自动补 0 所以大家都为 0
```

3、指定下标初始化

```
int arr[5]={[1]=30, [3]=40}; //0 30 0 40 0 将第 1 个元素初始化为 30 第 3 个元素初始化为 40
```

3、一维数值数组的操作

本质是对数组元素的操作。

数值数组 必须逐个元素操作。（记）

```
void test03()
{
    int arr[5]={0};
    int n = sizeof(arr)/sizeof(arr[0]);

    //数组的每个元素 等价 普通变量
    //num=100
    arr[0]=100;
    //num++
    arr[0]++; //arr[0]=arr[0]+1
    //data=num
    arr[1]=arr[0];

    //scanf("%d", &num)
    scanf("%d", &arr[2]);

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



```
2.10.9.42.114
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work/c/day02$ gcc 01_code.c
edu@edu:~/work/c/day02$ ./a.out
300
101 101 300 0 0
edu@edu:~/work/c/day02$
```

键盘给一维**数值数组**获取输入。

必须逐个元素 获取输入。（记）

```

void test04()
{
    int arr[5]={0};
    int n = sizeof(arr)/sizeof(arr[0]);

    printf("请输入%d个int数据:", n);
    int i=0;
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]); //默认每个数据以空格隔开
    }

    int sum=0;
    for(i=0;i<n;i++)
    {
        sum+=arr[i];
    }
    printf("数组元素的综合为:%d\n", sum);
}

```

```

2.10.9.42.114
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work/c/day02$ gcc 01_code.c
edu@edu:~/work/c/day02$ ./a.out
请输入5个int数据:10 20 30 40 50
数组元素的综合为:150
edu@edu:~/work/c/day02$

```

案例 1: 键盘输入 10 个 int 数, 求平均值

```

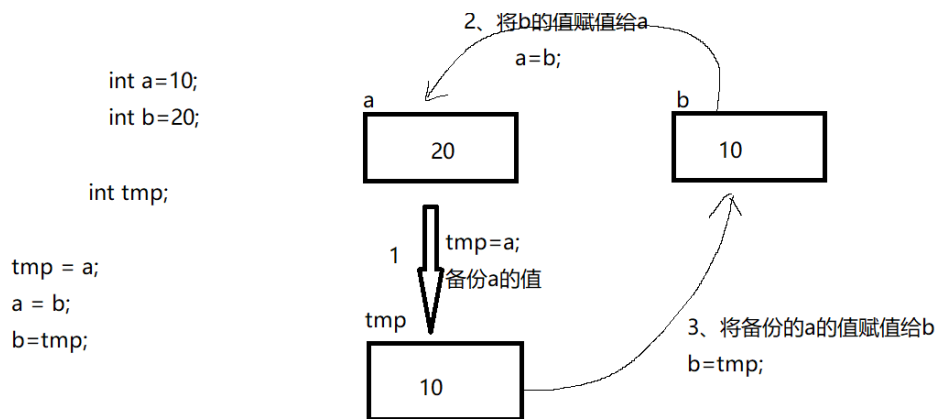
void test01()
{
    int arr[10] = {0};
    int n = sizeof(arr) / sizeof(int);

    printf("请输入%d个int数值:", n);
    int i = 0;
    for (i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }
    int sum = 0;
    for (i = 0; i < n; i++)
    {
        sum += arr[i];
    }
    printf("平均值为:%lf\n", (double)sum / n);

    return;
}

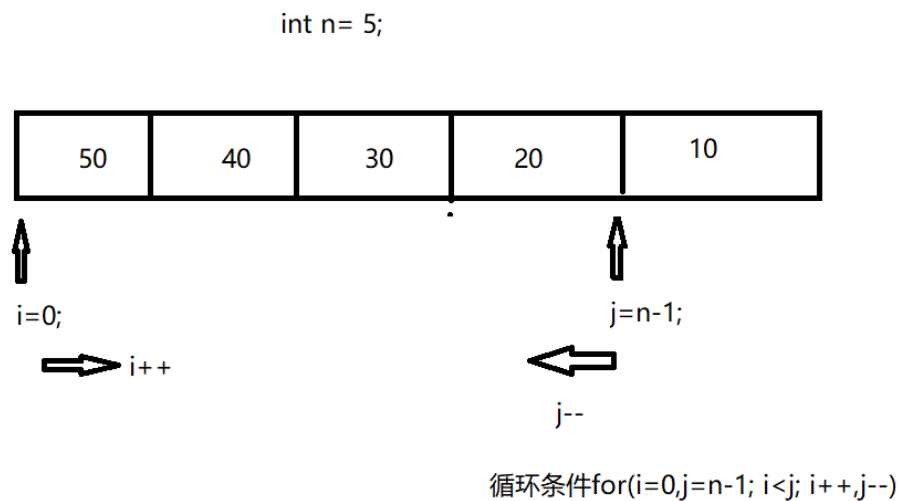
```

案例 2: 交换两个普通变量的值



案例 3（晚上案例） 键盘输入 10 个 int 数值 数组逆序（前后颠倒）。

比如输入：1 2 3 4 5 逆序后：5 4 3 2 1



交换代码：

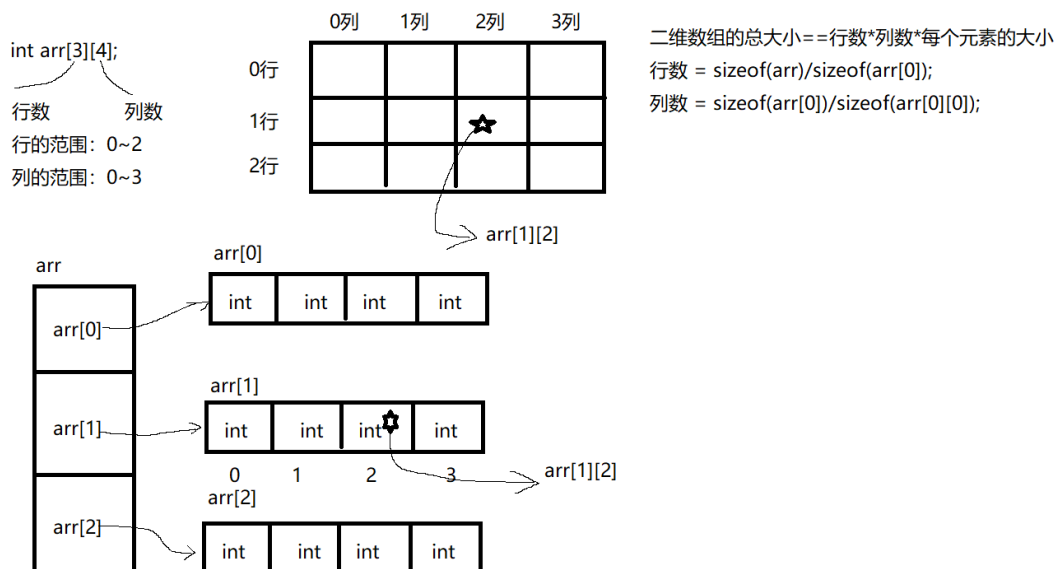
```

int tmp = arr[i];
arr[i] = arr[j];
arr[j] = tmp;

```

知识点 4 【二维数值数组】

1、二维数值数组的概述



2、二维数值数组的初始化

1、分段初始化（一行行初始化）

```
//分段初始化（全部初始化）  
  
int arr[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};  
  
//如果是全部初始化 可以省略二维数组的行数  
  
int arr[][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};  
  
int arr[3][4]={{1,2},{5},{9}};
```

2、连续初始化（放满一行 才放下一行）

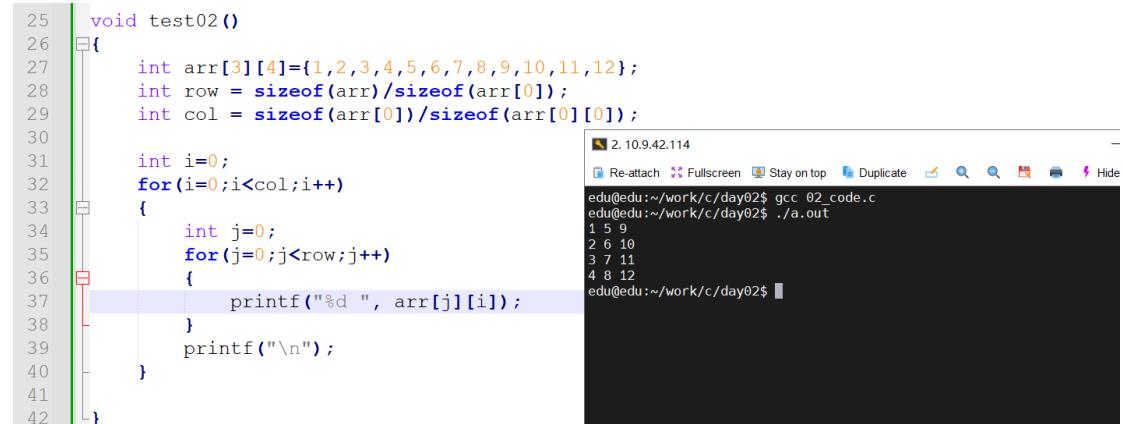
```
//连续初始化（全部初始化）  
  
//int arr[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};  
  
//如果是全部初始化 可以省略二维数组的行数  
  
//int arr[][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```



```
int arr[3][4]={1,2,5,9};
```

3、推荐的二维数值数组的初始化为 0 的方式

```
int arr[3][4]={0};
```



```
25 void test02()
26 {
27     int arr[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};
28     int row = sizeof(arr)/sizeof(arr[0]);
29     int col = sizeof(arr[0])/sizeof(arr[0][0]);
30
31     int i=0;
32     for(i=0;i<col;i++)
33     {
34         int j=0;
35         for(j=0;j<row;j++)
36         {
37             printf("%d ", arr[j][i]);
38         }
39         printf("\n");
40     }
41 }
42 }
```

Terminal output:

```
edu@edu:~/work/c/day02$ gcc 02_code.c
edu@edu:~/work/c/day02$ ./a.out
1 5 9
2 6 10
3 7 11
4 8 12
edu@edu:~/work/c/day02$
```

4、获取键盘输入

```
void test02()
{
    int arr[3][4]={0};

    int row = sizeof(arr)/sizeof(arr[0]);

    int col = sizeof(arr[0])/sizeof(arr[0][0]);

    printf("请输入%d 个 int 数值:",row*col);

    int i=0,j=0;

    for(i=0;i<row;i++)
    {
        for(j=0;j<col;j++)
        {
            scanf("%d", &arr[i][j]);
        }
    }
}
```

```

    }

}

for(i=0;i<row;i++)
{
    for(j=0;j<col;j++)
    {
        printf("%d ", arr[i][j]);

    }

    printf("\n");
}

}

```

案例 1：定义一个 5 行 4 列的二维数值数组，键盘获取输入，求出每一行的平均数据

```

int arr[5][4]={0};

int avg[5]={0};

void test02()

{

int arr[5][4]={0};

int row = sizeof(arr)/sizeof(arr[0]);

int col = sizeof(arr[0])/sizeof(arr[0][0]);

```

```
printf("请输入%d 个 int 数值:",row*col);
```

```
int i=0,j=0;
```

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

```
{
```

```
for(j=0;j<col;j++)
```

```
{
```

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

```
}
```

```
}
```

```
float avg[5]={0};
```

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

```
{
```

```
//sum 求一行的总和
```

```
int sum = 0;
```

```
for(j=0;j<col;j++)
```

```
{
```

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

```
}
```

```
avg[i] = (float)sum/col;
```

```

}

for(i=0;i<row;i++)

{

printf("%.2f ", avg[i]);

}

printf("\n");

}

```

知识点 5 【一维字符数组】

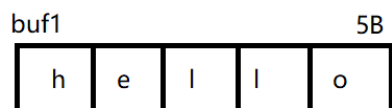
1、一维字符数组

一维字符数组：存放字符串

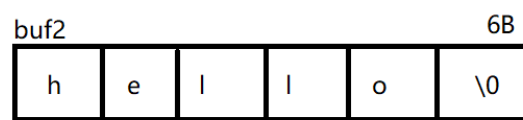
```
char buf[128];
```

2、字符数组的初始化方式(背)

```
char buf1[]={'h','e','l','l','o'};
```



```
char buf2[]="hello";
```



3、字符数组的遍历

```

void test02()

{

//以字符串的方式初始化 字符串的末尾默认有'\0'

char buf[128]="hello world";

```

//逐个字符遍历(需要遍历很多无效的数据)

```
int i=0;

for(i=0;i<128;i++)

{

    printf("%c", buf[i]);

}

printf("\n");
```

//逐个元素遍历 遇到'\0'自动结束遍历 如果需要逐个元素操作（遍历、拷贝、查询、追加）

```
i=0;

while(buf[i] != '\0')

{

    printf("%c",buf[i]);

    i++;

}

printf("\n");
```

//%s 遍历一个字符串 遇到'\0'自动结束输出 %s 需要的是字符串的首元素地址（背）
(推荐)

```
printf("%s\n",buf);
```

```
}
```

需要操作字符数组的每一个元素 使用 while，如果仅仅遍历字符数组 使用%s

%s 必须给定寿元地址 遇到'\0'结束

```
void test03()
{
    char buf1[128]="hello world";
    printf("%s\n",buf1);//hello world

    char buf2[128]="hello\0world";
    printf("%s\n",buf2);//hello

    char buf3[128]="\0hello\0world";
    printf("%s\n",buf3);//无输出

    int i=0;
    for(i=0;i<128;i++)
    {
        printf("%c", buf3[i]);
    }

    printf("\n");

    //如果只是查看字符串的内容可以选择%s
```

```

//如果需要逐个元素操作字符 遇到'\0'结束 while(buf[i] != '\0')

//如果希望遍历字符数组每个元素，不管是不是'\0',需要用 for

}

```

4、一维字符数组的元素操作

```

void test04()

{

    char buf[128]="hello world";

    printf("%d %c\n", buf[4], buf[4]);//111 o

    buf[8] = buf[8]-32;

    printf("%s\n",buf);//hello woRld

    printf("%s\n", &buf[2]);//llo world

}

```

5、键盘获取字符串

1、%s 不能获取带空格的字符串(重要)

```

void test05()
{
    //可以将整个字符数组初始化为0
    char buf[128]="";

    printf("请输入一个字符串:");
    //scanf获取字符串 遇到空格、回车自动结束 %s不能获取带空格的字符串
    scanf("%s", buf);

    printf("%s\n",buf);
}

int main(int argc,char *argv[])
{
    test05();
}

```

2. 10.9.42.114

Re-attach Fullscreen Stay on top Duplicate

```

edu@edu:~/work/c/day02$ gcc 03_code.c
edu@edu:~/work/c/day02$ ./a.out
请输入一个字符串:hello world
hello
edu@edu:~/work/c/day02$

```

2、gets 获取带空格的字符串（危险！！！！）

```

void test06()
{
    //可以将整个字符数组初始化为0
    char buf[6]="";

    printf("请输入一个字符串:");
    //gets获取带空格的字符串 放入buf中
    //gets 获取字符串的时候 不会判断存储空间是否越界 很容易访问非法内存
    gets(buf);//危险

    printf("###%s###\n",buf);
}

```

```

edu@edu:~/work/c/day02$ gcc 03_code.c
03_code.c: In function 'test06':
03_code.c:94:2: warning: implicit declaration of function 'gets' [enabled=warning]
    gets(buf);
    ^
/tmp/cc8YtmRh.o: 在函数'test06'中:
03_code.c:(.text+0x45b): 警告: the 'gets' function is dangerous and should not be used.
edu@edu:~/work/c/day02$ ./a.out
请输入一个字符串:123456789 987654321
##123456789 987654321##
*** stack smashing detected ***: ./a.out terminated
已放弃 (核心已转储)
edu@edu:~/work/c/day02$

```

3、fgets 获取带空格的字符串（推荐，重要）

```
char *fgets(char *s, int size, FILE *stream);
```

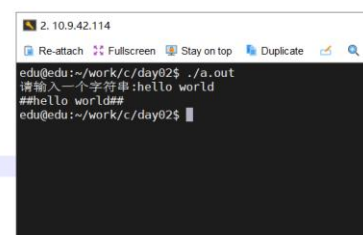
参数 s: 表示存储字符串的空间起始地址

参数 size: 表示最大能获取字符串的字节数-1

参数 stream: 输入设备 默认 stdin 标准输入设备（键盘）

返回值: 字符串的空间起始地址 也就是 s 指向的地址

```
100 #include<string.h>
101 void test07()
102 {
103     //可以将整个字符数组初始化为0
104     char buf[128]="";
105
106     printf("请输入一个字符串:");
107     //能获取带空格的字符串，遇到换行符结束 但是要获取换行符
108     fgets(buf, sizeof(buf), stdin);
109     buf[strlen(buf)-1]=0;
110
111     printf("###s###n", buf);
112 }
```



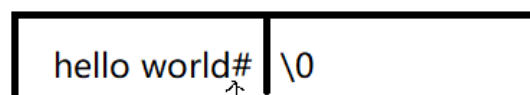
```
char buf[128]="";
```

```
fgets(buf, sizeof(buf), stdin);
```

```
buf[strlen(buf)-1]=0;
```

给buf中的回车符 赋值成\0

buf



纪录回车符的下标

```
strlen(buf)-1
```

4、sizeof 和 strlen 的区别

sizeof 是测量类型的大小。

strlen 是测量字符串的长度（遇到'\0'结束不包含'\0'） 需要头文件: #include <string.h>


```

void test08 ()
{
    char buf[128]="hello world";
    printf("%ld\n",sizeof(buf)); //128
    printf("%d\n", strlen(buf)); //11

    char buf1[]="hello world";
    printf("%ld\n",sizeof(buf1)); //12
    printf("%d\n", strlen(buf1)); //11

    char buf2[]="hello\0world";
    printf("%ld\n",sizeof(buf2)); //12
    printf("%d\n", strlen(buf2)); //5

    char buf3[]={ 'h','e','l','l','o' };
    printf("%ld\n",sizeof(buf3)); //5
    printf("%d\n", strlen(buf3)); //未知
}

```

6、八进制转义和十六进制转义字符

1、八进制转义

\ddd 三个 d 表示最多识别 3 位 每个 d 的范围是 0~7.

```
'\123' '\57' '\58'error '\1234' error
```

'\123'的 ASCII 值为_83_(将\123 看成八进制数 123 将其转换成十进制数 就是他的 ASCII 值)

```
char buf[]="hello\0\123\127\181world";
```

```
printf("%ld\n",sizeof(buf)); //17
```

```
printf("%d\n", strlen(buf)); //5
```

2、十六进制转义

\xhh 每个 h 的范围 0~9 a~f

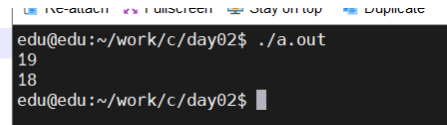
```
char buf[]="hello\xabcde\xfgworld";
```

```
printf("%ld\n", sizeof(buf)); //14
```

```
printf("%d\n", strlen(buf)); //13
```

\t 是 tab \r 回到行首 \n 换行符 \\ 才代表字符 %% 表示一个%号

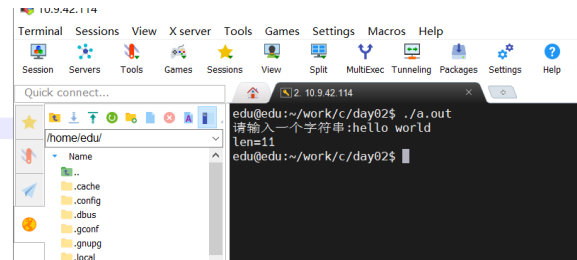
```
char buf[]="%%hello\r\\123\x128\578\cbc";
printf("%ld\n", sizeof(buf)); //19
printf("%d\n", strlen(buf)); //18
```



```
edu@edu:~/work/c/day02$ ./a.out
19
18
edu@edu:~/work/c/day02$
```

案例 1 (晚上) : char buf1[128], 键盘输入字符串 不允许使用库函数 实现 strlen 的功能

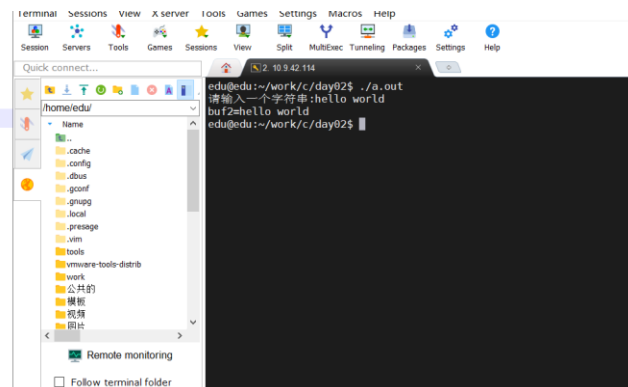
```
148 void test11()
149 {
150     char buf[128]="";
151     printf("请输入一个字符串:");
152     fgets(buf, sizeof(buf), stdin);
153     buf[strlen(buf)-1]='\0';
154
155     int len=0;
156     while(buf[len] != '\0')
157         len++;
158     printf("len=%d\n", len);
159 }
```



```
edu@edu:~/work/c/day02$ ./a.out
请输入一个字符串:hello world
len=11
edu@edu:~/work/c/day02$
```

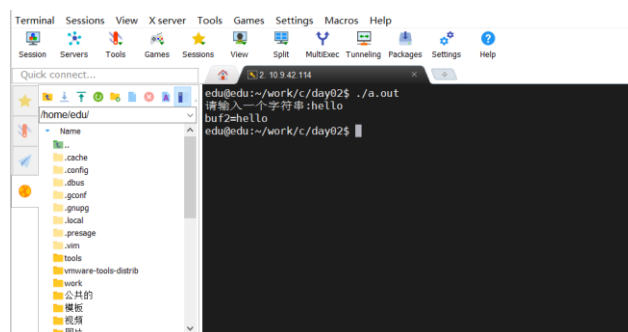
案例 2 (晚上) : char buf1[128], char buf2[128] 键盘输入字符串存入 buf1 中 不允许使用库函数 实现将 buf1 的字符串 拷贝到 buf2 中

```
161 void test12()
162 {
163     char buf1[128]="";
164     printf("请输入一个字符串:");
165     fgets(buf1, sizeof(buf1), stdin);
166     buf1[strlen(buf1)-1]='\0';
167
168     char buf2[128]="";
169
170     int i=0;
171     while(buf1[i] != '\0')
172     {
173         buf2[i]=buf1[i];
174         i++;
175     }
176     buf2[i]='\0';
177
178     printf("buf2=%s\n", buf2);
179 }
180
```



```
edu@edu:~/work/c/day02$ ./a.out
请输入一个字符串:hello world
buf2=hello world
edu@edu:~/work/c/day02$
```

```
161 void test12()
162 {
163     char buf1[128]="";
164     printf("请输入一个字符串:");
165     fgets(buf1, sizeof(buf1), stdin);
166     buf1[strlen(buf1)-1]='\0';
167
168     char buf2[128]="";
169
170     int i=0;
171     while((buf2[i] = buf1[i]) && ++i);
172
173     buf2[i]='\0';
174
175     printf("buf2=%s\n", buf2);
176 }
177
```



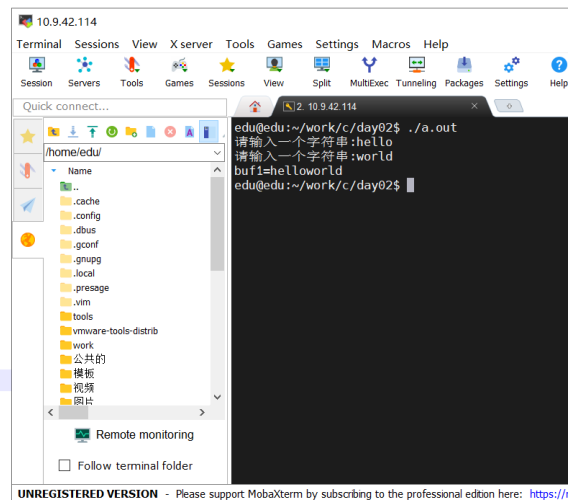
```
edu@edu:~/work/c/day02$ ./a.out
请输入一个字符串:hello
buf2=hello
edu@edu:~/work/c/day02$
```

案例 3 (晚上) : char buf1[128], char buf2[128] 键盘输入两个字符串分别存入 buf1, buf2 中 不允许使用库函数 实现将 buf2 追加到 buf1 的尾部

```

179 void test13()
180 {
181     char buf1[128]="";
182     printf("请输入一个字符串:");
183     fgets(buf1,sizeof(buf1),stdin);
184     buf1[strlen(buf1)-1]='\0';
185
186     char buf2[128]="";
187     printf("请输入一个字符串:");
188     fgets(buf2,sizeof(buf2),stdin);
189     buf2[strlen(buf2)-1]='\0';
190
191     //定位buf1的尾部
192     int end=0;
193     while(buf1[end]!='\0')
194         end++;
195
196     //将buf2的逐个元素 赋值到buf1的尾部之后
197     int i=0;
198     while(buf2[i] != '\0')
199     {
200         buf1[end]=buf2[i];
201         end++;
202         i++;
203     }
204     buf1[end]='\0';
205     printf("buf1=%s\n",buf1);
206 }
207

```



案例 4: 比如有如下两个字符串, 不能使用库函数 完成字符串的比较

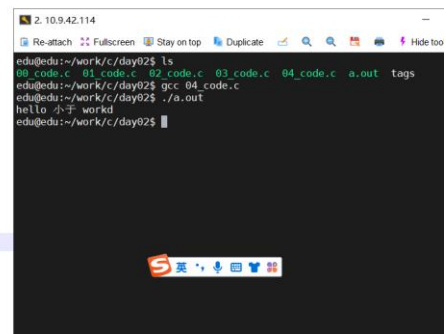
```
char buf1[128]="hello";
```

```
char buf2[128]="workd";
```

```

1  #include<stdio.h>
2  void test01()
3  {
4      char buf1[128]="hello";
5      char buf2[128]="workd";
6
7      //逐个元素比较 相等才比较下一个 完全相等才相等
8      int i=0;
9      int flag = 0;
10     while( !(flag = buf1[i]-buf2[i]) && (buf1[i]!='\0') && ++i );
11
12     if(flag > 0)
13     {
14         printf("%s 大于 %s\n",buf1,buf2);
15     }
16     else if(flag < 0)
17     {
18         printf("%s 小于 %s\n",buf1,buf2);
19     }
20     else if(flag == 0)
21     {
22         printf("%s 相等 %s\n",buf1,buf2);
23     }
24 }

```



案例 5:键盘输入一个字符串判断 它是否是回文。

```
12321
```

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

```
void test02()
```

```
{
```

```
char buf[128]="";
```

```
printf("请输入一个字符串:");
```

```
fgets(buf,sizeof(buf),stdin);
```

```
buf[strlen(buf)-1]='\0';
```

```
//判断是否是回文
```

```
//将 end 定位到尾元素
```

```
int end=0;
```

```
while(buf[end] && ++end);
```

```
--end;
```

```
//首尾元素 相等才比较下一个
```

```
int i=0;
```

```
while(buf[i]==buf[end])
```

```
{
```

```
i++;
```

```
end--;
```

```
if(i>=end)
```

```
break;
```

```
}
```

```
if(i>=end)
```

```
{
```

```

printf("%s 是回文\n",buf);

}

else

{

printf("%s 不是回文\n",buf);

}

}

```

案例 6：键盘输入字符串 buf1， 输入插入的位置 pos,在输入字符串 buf2, 需求：将 buf2 插入到 buf1 的 pos 位置

```
char buf1[128]="hello"
```

```
char buf2[128]="world";
```

```
int pos = 3;
```

```
"helworldlo"
```

```

void test03()

{

char buf1[128]="";

printf("请输入一个字符串:");

fgets(buf1,sizeof(buf1),stdin);

buf1[strlen(buf1)-1]='\0';

//得到 buf1 的字符串长度

```

```
int n=0;

while(buf1[n] && ++n);

printf("请输入插入的位置:");

int pos = 0;

scanf("%d",&pos);

//去掉缓冲区回车

getchar();


//判断 pos 的合法

if(pos <0 || pos > n)

{

printf("位置%d 不合法\n",pos);

return;

}


char buf2[128]="";

printf("请输入一个字符串:");

fgets(buf2,sizeof(buf2),stdin);

buf2[strlen(buf2)-1]='\0';


//得到 buf2 的长度
```

```
int len = 0;

while(buf2[len] && ++len);

//buf1 预留 len 的空间

int i=0;

for(i=n; i>=pos; i--)

{

    buf1[i+len] = buf1[i];

}

//将 buf2 逐个元素 放入 buf1 的预留位置

i=0;

while(buf2[i] != '\0')

{

    buf1[pos] = buf2[i];

    i++;

    pos++;

}

printf("插入后的结果:%s\n", buf1);

}
```

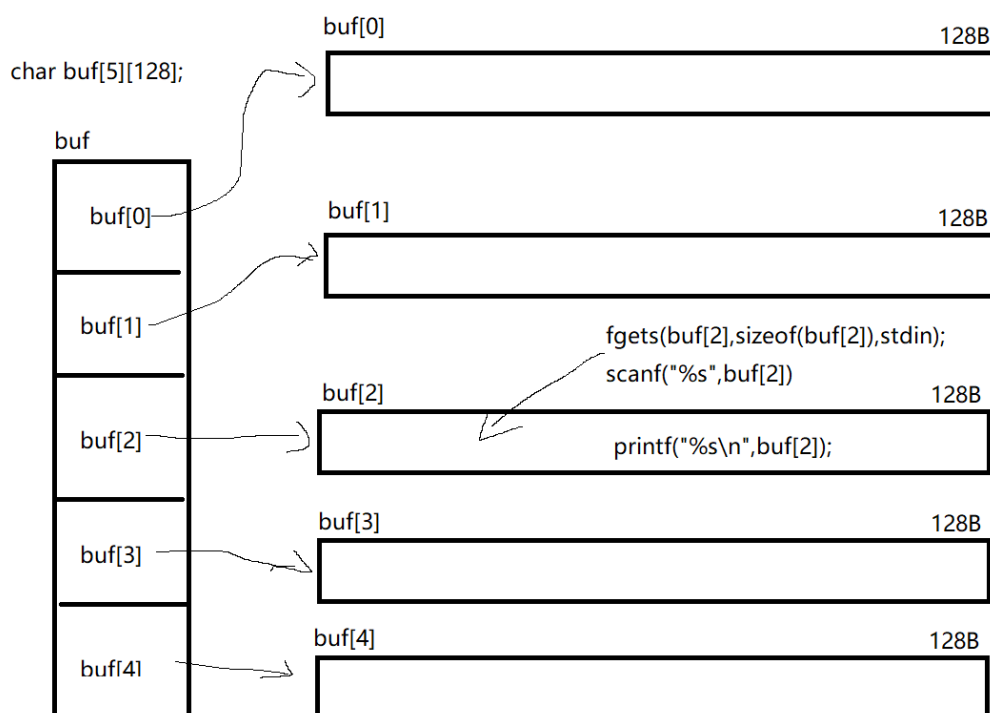
```
edu@edu:~/work/c/day02$ gcc 04_code.c
edu@edu:~/work/c/day02$ ./a.out
请输入一个字符串:hello
请输入插入的位置:3
请输入一个字符串:world
插入后的结果:helworldlo
edu@edu:~/work/c/day02$
```

知识点 6 【二维字符数组】

一维字符数组：本质存放一个字符串

```
char buf[5][128]= "" ;
```

二维字符数组：本质存放多个字符串，每一行存放一个字符串。




```

112 void test04()
113 {
114     //初始化(连续初始化)
115     //char buf[5][128]={'h','e','l','l','o'};
116     //初始化(分段初始化)
117     //char buf[5][128]={{'h','e'},{'l','l'}};
118     //初始化(推荐, 每一行以字符串的方式初始化)
119     char buf[5][128]={"hello","world","xixi","hehehe","hahaha"};
120     int row = sizeof(buf)/sizeof(buf[0]);
121     int i=0;
122     for(i=0;i<row;i++)
123     {
124         printf("%s\n",buf[i]);
125     }
126     printf("%c\n",buf[2][1]);
127 }
128

```

```

2. 10.9.42.114
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work/c/day02$ gcc 04_code.c
edu@edu:~/work/c/day02$ ./a.out
hello
world
xixi
hehehe
hahaha
i
edu@edu:~/work/c/day02$

```

```

130 void test05()
131 {
132     char buf[5][128]={""};
133     int row = sizeof(buf)/sizeof(buf[0]);
134
135     printf("请输入%d个字符串\n",row);
136     int i=0;
137     for(i=0;i<row;i++)
138     {
139         //scanf("%s", buf[i]);
140         fgets(buf[i],sizeof(buf[i]),stdin);
141     }
142     for(i=0;i<row;i++)
143     {
144         printf("----%s\n", buf[i]);
145     }
146
147
148 }

```

```

2. 10.9.42.114
Re-attach Fullscreen Stay on top Duplicate
edu@edu:~/work/c/day02$ gcc 04_code.c
edu@edu:~/work/c/day02$ ./a.out
请输入5个字符串
hehhe xioxix lalal
sdfasdfsdfa
sfasdfsfasdfasf
sdfasdfsfa fasdfa fas afsdf
fasdfa afsdfa asdfa fa
----hehhe xioxix lalal
----sdfasdfsdfa
----sfasdfsfasdfasf
----sdfasdfsfa fasdfa fas afsdf
----fasdfa afsdfa asdfa fa
edu@edu:~/work/c/day02$

```