知识点1【字符串操作函数】（了解）

1、测量字符串的长度 strlen

2、字符串拷贝函数

3、字符串追加函数strcat

4、字符串比较strcmp

5、字符查找函数strchr

6、字符串查找strstr

7、字符串 转 数值

8、字符串 切割

案例1：第一种切割法

案例2：切割方式二

案例1：

知识点2【将字符串 转成 数值】（了解）

知识点3【格式化字符串】（了解）

1、sprintf 用于组包

案例1：将数值转成字符串

2、sscanf 用于解包

知识点4【sscanf 高级用法】（了解）

1、跳过数据%*d %*s

2、读指定宽度的数据：%[width]s %[width]d

案例1：有字符串"12345678"请将34用整数提取 67用字符串提取

3、%[a-z] 表示匹配 a 到 z 中任意字符(尽可能多的匹配)

知识点1【字符串操作函数】（了解）

以str开头的函数 都是字符串操作函数 都是遇到'\0'结束操作

1、测量字符串的长度 strlen

```
#include <string.h>
size_t strlen(const char *s);
s指 需要测量字符串的首元素地址
```

```
char str[128]="hello";
strlen(str) == 5;
```

2、字符串拷贝函数

```
#include <string.h>
char *strcpy(char *dest, const char *src);
char *strncpy(char *dest, const char *src, size_t n);
dest:目的空间地址
src: 原字符串的首元素地址
```

```c
#include <stdio.h>
#include <string.h>
void test01()
{
    char dst[128] = "";
    char src[] = "hello\0world";
    strcpy(dst, src);
    printf("%s\n", dst);

    char dst1[] = "";
    char src1[] = "helloworld";

    strcpy(dst1, src1); //dst1只有1字节 拷贝越界
    printf("%s\n", dst1);
    printf("%s\n", src1);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
hello
helloworld
elloworld
edu@edu:~/work/c/day06$
```

3、字符串追加函数strcat

```c
#include <string.h>
char *strcat(char *dest, const char *src);
char *strncat(char *dest, const char *src, size_t n);
```

将src指向的字符串 追加到 dest指向的字符串尾部

```c
void test02()
{
    char dst[128] = "hello";
    char src[] = "world";
    strcat(dst, src);
    printf("%s\n", dst);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
helloworld
edu@edu:~/work/c/day06$
```

4、字符串比较strcmp

```c
#include <string.h>
int strcmp(const char *s1, const char *s2);
int strncmp(const char *s1, const char *s2, size_t n);
```

返回值：

>0　s1字符串 > s2字符串

<0　s1字符串 < s2字符串

==0　s1字符串==s2字符串

```c
void test03()
{
    char s1[128] = "";
    char s2[128] = "";

    printf("请输入第一个字符串:");
    scanf("%s", s1);
```

```c
    printf("请输入第二个字符串:");
    scanf("%s", s2);

     if (strcmp(s1, s2) > 0)
     {
         printf("%s 大于 %s\n", s1, s2);
     }
     else if (strcmp(s1, s2) < 0)
     {
         printf("%s 小于 %s\n", s1, s2);
     }
     else if (strcmp(s1, s2) == 0)
     {
         printf("%s 等于 %s\n", s1, s2);
     }
}
```

```
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
请输入第一个字符串:hello
请输入第二个字符串:hello
hello 等于 hello
edu@edu:~/work/c/day06$ ./a.out
请输入第一个字符串:hello
请输入第二个字符串:hehe
hello 大于 hehe
edu@edu:~/work/c/day06$ ./a.out
请输入第一个字符串:hello
请输入第二个字符串:hellop
hello 小于 hellop
edu@edu:~/work/c/day06$
```

5、字符查找函数strchr

```c
#include <string.h>
char *strchr(const char *s, int c);
char *strrchr(const char *s, int c);
strchr从前往后找 第一次出现c的地址，如果没找到 返回NULL
```

```c
void test04()
{
    char src[] = "hello world";

    char *ret = strchr(src, 'o');
    if (ret != NULL)
    {
        printf("%s\n", ret);
    }
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
o world
edu@edu:~/work/c/day06$
```

6、字符串查找strstr

```c
#include <string.h>
char *strstr(const char *haystack, const char *needle);
返回值：找到返回找到的地址 失败 返回NULL
```

```c
void test04()
{
    char src[] = "http://www.sex.777.sex.999.sex.com";

    char *ret = strstr(src, "sex");
    if (ret != NULL)
    {
        printf("%s\n", ret);
    }
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
[sudo] edu 的密码：
edu@edu:~/work/c/day06$ ./a.out
sex.777.sex.999.sex.com
edu@edu:~/work/c/day06$
```

```c
void test04()
{
    char src[] = "http://www.sex.777.sex.999.sex.com";

    printf("%s\n", src);
    while (1)
    {
        char *ret = strstr(src, "sex");
        if (ret == NULL)
            break;
        memset(ret, '*', strlen("sex"));
    }

    printf("%s\n", src);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
http://www.sex.777.sex.999.sex.com
http://www.***.777.***.999.***.com
edu@edu:~/work/c/day06$
```

7、字符串 转 数值

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

atoi将字符串 转成 int类型

atol将字符串 转成 long类型

atof将字符串 转成 float类型

```c
#include <stdlib.h>
void test05()
{
    printf("%d\n", atoi("100abc"));
    printf("%ld\n", atol("100abc"));
    printf("%f\n", atof("3.14f"));
}
int main(int argc, char const *argv[])
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
100
100
3.140000
edu@edu:~/work/c/day06$
```

8、字符串 切割

```c
#include <string.h>
char *strtok(char *str, const char *delim);
```

第一次切割：str必须指向 待切割的字符串的首元素地址　delim指向分割符":"

后续切割：str传NULL　delim指向分割符":"

返回值：

成功：返回值子串的首元素地址

失败：返回NULL

```c
"hehehe:xixixi:hahaha:lalala#heiheihei:henhenhen:wuwuwu"
```

案例1：第一种切割法

```c
void test06()
{
    char str[] = "hehehe:xixixi:hahaha:lalala:heiheihei:henhenhen:wuwuwu"
    char *buf[32]; //存放子串的首元素地址

    //第一次切
    int i = 0;
    buf[i] = strtok(str, ":");

     //后续切：上一次切割正常
     while (buf[i] != NULL)
     {
         i++;
         buf[i] = strtok(NULL, ":");
     }

     //遍历切割到的子串
     i = 0;
     while (buf[i] != NULL)
     {
         printf("%s\n", buf[i]);
         i++;
```

```
    }
}
```

```
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
edu@edu:~/work/c/day06$ ./a.out
hehehe
xixixi
hahaha
lalala
heiheihei
henhenhen
wuwuwu
edu@edu:~/work/c/day06$
```

案例2：切割方式二

```c
void test06()
{
    char str[] = "hehehe:xixixi:hahaha:lalala:heiheihei:henhenhen:wuwuwu"
    char *buf[32] = {str}; //存放子串的首元素地址
    //buf[0] = str

    //后续切：上一次切割正常
    int i = 0;
    while (1)
    {
        buf[i] = strtok(buf[i], ":");
        if (buf[i] == NULL)
            break;
        i++;
    }
}
```

```c
void test06()
{
    char str[] = "hehehe:xixixi:hahaha:lalala:heiheihei:henhenhen:wuwuwu";
    char *buf[32] = {str}; //存放子串的首元素地址

    int i = 0;
    while ((buf[i] = strtok(buf[i], ":")) && ++i);

    //遍历切割到的子串
    i = 0;
    while (buf[i] != NULL)
    {
        printf("%s\n", buf[i]);
        i++;
    }
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc
edu@edu:~/work/c/day06$ ./a.out
hehehe
xixixi
hahaha
lalala
heiheihei
henhenhen
wuwuwu
edu@edu:~/work/c/day06$
```

案例1：

```c
char msg_src[]="+CMGR:REC UNREAD,+8613466630259,98/10/01,18:22:11+00,ABCdefGHI"
```

以下为我们的手机收到的短信的格式，请利用指针数组与 strtok 函数对其解析

char msg_src[]="+CMGR:REC UNREAD,+8613466630259,98/10/01,18:22:11+00,ABCdefGHI";

参考以下的函数名字以及参数，完成相应的要求

int msg_deal(char *msg_src, char *msg_done[],char *str)

参数 1：待切割字符串的首地址

参数 2：指针数组：存放切割完字符串的首地址

参数 3：切割字符

返回值：切割的字符串总数量

```c
int msg_deal(char *msg_src, char *msg_done[], char *str)
{
    int i = 0;
    while ((msg_done[i] = strtok(msg_done[i], str)) && (++i))
        ;

    return i;
}
void test07()
{
    char msg_src[] = "+CMGR:REC UNREAD,+8613466630259,98/10/01,18:22:11+00,ABCdefGHI";
    char *msg_done[32] = {msg_src};
    int num = 0;

```

```c
    num = msg_deal(msg_src, msg_done, ",");

    printf("字符串的数量:%d\n", num);

    int i = 0;
    while (msg_done[i] != NULL)
    {
        printf("%s\n", msg_done[i++]);
    }
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 01_code.c
[sudo] edu 的密码:
edu@edu:~/work/c/day06$ ./a.out
字符串的数量:5
+CMGR:REC UNREAD
+8613466630259
98/10/01
18:22:11+00
ABCdefGHI
edu@edu:~/work/c/day06$
```

知识点2【将字符串 转成 数值】（了解）

```c
int my_atoi(char *str)
{
    int sum = 0;
    while (*str >= '0' && *str <= '9')
    {
        sum = sum * 10 + (*str - '0');
        str++;
    }

    return sum;
}
void test08()
{
    char buf[] = "12345abc";

    printf("%d\n", my_atoi(buf));
}
```

```
edu@edu: ~/work/c/day06
12345
edu@edu:~/work/c/day06$
```

知识点3【格式化字符串】（了解）

组包：按照需要的格式 组成字符串

解包：解析特定格式的数据

1、sprintf 用于组包

将零散的数据 按照固定的格式 组成字符串

```
#include <stdio.h>
int sprintf(char *str, const char *format,...);
sprintf返回值为实际组包的长度
```

```
void test01()
{
    int year = 2021;
    int month = 7;
    int day = 30;

    char buf[128] = "";
    int len = sprintf(buf, "%d年%d月%d日", year, month, day);
    printf("len=%d, buf=%s\n", len, buf);
}
int main(int argc, char const *argv[])
{
    test01();
    return 0;
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ ./a.out
len=16, buf=2021年7月30日
edu@edu:~/work/c/day06$
```

案例1：将数值转成字符串

```
void test02()
{
    char buf[128] = "";
    sprintf(buf, "%d", 1234);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=1234
edu@edu:~/work/c/day06$
```

2、sscanf 用于解包

%d提取数值 '0'~'9'

```
void test03()
{
    char buf[128] = "2021年7月30日";
    int year = 0;
    int month = 0;
    int day = 0;

    //sscanf 和 %d 提取'0'~'9'
    sscanf(buf, "%d年%d月%d日", &year, &month, &day);
    printf("%d %d %d\n", year, month, day);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
2021 7 30
edu@edu:~/work/c/day06$
```

%s提取字符串 遇到'\0' 空格 回车

```c
void test04()
{
    char buf[128] = "2021年 7月 30日";
    char msg[128] = "";
    sscanf(buf, "%s", msg);
    printf("msg=%s\n", msg);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
msg=2021年
edu@edu:~/work/c/day06$
```

知识点4【sscanf 高级用法】（了解）

1、跳过数据%*d %*s

```c
sscanf("1234 5678", "%*d %s", buf);//buf="5678"
```

```c
void test05()
{
    char buf[128] = "";
    sscanf("1234::::5678", "%*d:%s", buf);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=::::5678
edu@edu:~/work/c/day06$
```

2、读指定宽度的数据：%[width]s %[width]d

```c
void test06()
{
    char buf[128] = "";
    sscanf("12345678", "%3s", buf);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=123
edu@edu:~/work/c/day06$
```

```c
void test06()
{
    int num = 0;
    sscanf("12345678", "%3d", &num);
    printf("num=%d\n", num);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ ./a.out
num=123
edu@edu:~/work/c/day06$
```

案例1：有字符串"12345678"请将34用整数提取 67用字符串提取

```c
void test06()
{
    int num = 0;        //34
    char buf[128] = ""; //67
    sscanf("12345678", "%*2s%2d%*c%2s", &num, buf);
    printf("num=%d, buf=%s\n", num, buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
[sudo] edu 的密码:
edu@edu:~/work/c/day06$ ./a.out
num=34, buf=67
edu@edu:~/work/c/day06$
```

3、%[a-z] 表示匹配 a 到 z 中任意字符(尽可能多的匹配)

```c
void test06()
{
    char buf[128] = "";
    sscanf("abcABCde", "%[a-z]", buf);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=abc
edu@edu:~/work/c/day06$
```

```c
void test06()
{
    char buf[128] = "";
    sscanf("abcABCde123", "%[a-z,A-Z,0-9]", buf);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=abcABCde123
edu@edu:~/work/c/day06$
```

4、%[aBc] 匹配 a、B、c 中一员，贪婪性

```c
void test06()
{
    char buf[128] = "";
    sscanf("abcABCde123", "%[aBc]", buf);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=a
edu@edu:~/work/c/day06$
```

5、%[^aFc] 匹配非 a Fc 的任意字符，贪婪性

```c
void test06()
{
    char buf[128] = "";
    sscanf("abcABCde123", "%[^Bc]", buf);
    printf("buf=%s\n", buf);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
buf=ab
edu@edu:~/work/c/day06$
```

```c
void test06()
{
    char buf[128] = "[02:04.94][00:36.09]我想大声宣布 对你依依不舍";

    char *song_lrc = buf;
    //定位到歌词的位置
    while (*song_lrc == '[')
    {
        song_lrc += 10;
    }

    //逐个时间分析
    char *time_lrc = buf;
    while (*time_lrc == '[')
    {
```

```c
        int m = 0;
        int s = 0;
        sscanf(time_lrc, "[%d:%d", &m, &s);
        printf("时间%d秒 唱歌词:%s\n", m * 60 + s, song_lrc);

        time_lrc += 10;
    }
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
时间124秒 唱歌词:我想大声宣布 对你依依不舍
时间36秒 唱歌词:我想大声宣布 对你依依不舍
edu@edu:~/work/c/day06$
```

```c
void test07()
{
    char buf[128] = "lianghe@1000phone.com";
    char name[128] = "";
    char log[128] = "";

    sscanf(buf, "%[^@]@%[^.]", name, log);
    printf("name=%s, log=%s\n", name, log);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
name=lianghe, log=1000phone
edu@edu:~/work/c/day06$
```

```c
#include <string.h>
void test08()
{
    char buf[] = "+CMGR:REC UNREAD,+8613466630259,98/10/01,18:22:11+00,ABCdefGHI";
    char *msg[32] = {buf};

    int i = 0;
    while ((msg[i] = strtok(msg[i], ",")) && (++i))
        ;

     //短信的读取状态
     //msg[0] = "+CMGR:REC UNREAD"
     char status[128] = "";
```

```c
    sscanf(msg[0], "%*s %s", status);
    if (strcmp(status, "UNREAD") == 0)
    {
        printf("有未读信息\n");
    }
    else if (strcmp(status, "READ") == 0)
    {
        printf("已读信息\n");
    }

    //msg[1]="+8613466630259"
    printf("手机号:%s\n", msg[1] + 3);

    //msg[2]="98/10/01"
    int year = 0;
    int month = 0;
    int day = 0;
    sscanf(msg[2], "%d/%d/%d", &year, &month, &day);
    printf("日期:%d年%02d月%02d日\n", year + 1900, month, day);

    //msg[3]="18:22:11+00"
    int h = 0, m = 0, s = 0;
    sscanf(msg[3], "%d:%d:%d", &h, &m, &s);
    printf("时间:%02d时%02d分%02d秒\n", h, m, s);

    printf("收到的消息:%s\n", msg[4]);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 02_code.c
edu@edu:~/work/c/day06$ ./a.out
有未读信息
手机号:13466630259
日期:1998年10月01日
时间:18时22分11秒
收到的消息:ABCdefGHI
edu@edu:~/work/c/day06$
```

知识点5【const】（了解）

1、const修饰普通变量 为只读变量

```c
void test01()
{
    //num为只读  只能初始化 不能被赋值
    const int num = 10;
    // num = 100; //err
    printf("num = %d\n", num);
}
```

2、const 修饰*

```
const int *p
在使用中：
 *p是只读 不同通过*p 修改p所指向的空间内容
 p 可读可写 p可以指向其他空间
```

```c
void test02()
{
    int num = 10;
    //*p只读  p可读可写
    const int *p = &num;
    //*p = 100; //error

    int data = 20;
    p = &data;
    printf("*p = %d\n", *p);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ ./a.out
*p = 20
edu@edu:~/work/c/day06$
```

3、const 修饰 指针变量

```
int * const p = &num;
p只读 除了初始化 不能修改p的指向
*p可读可写 可以通过*p修改p指向的空间内容
```

```c
void test02()
{
    int num = 10;
    //p只读  *p可读可写
    int *const p = &num;
    *p = 100;

    int data = 20;
    //p = &data; //error
    printf("*p = %d\n", *p);
}
```

```
edu@edu: ~/work/c/day06
edu@edu:~/work/c/day06$ sudo gcc 03_code.c
edu@edu:~/work/c/day06$ ./a.out
*p = 100
edu@edu:~/work/c/day06$
```

4、const 既修饰* 也修饰指针变量

```c
const int *const p
```

```c
void test02()
{
    int num = 10;
    //p只读  *p只读
    const int *const p = &num;
    //*p = 100;//error

    int data = 20;
    //p = &data; //error
    printf("*p = %d\n", *p);
}
```