strstr

```
strstr ( 在一字符串中查找指定的字符串 )
 头文件:
 #include <string.h>
 定义函数:
 char *strstr(const char *haystack, const char * needle);
 参数分析:
 haystack --> 需要查找的源字符串 （ "Hello Even" ）
 needle --> 需要查找的字符串 ("Even") 查找到的内容
 返回值:
  成功 返回第一次出行的地址
  失败 返回0 NULL
```

strlen

```
strlen ( 返回字符串长度 )
 头文件:
 #include <string.h>
 定义函数:
 size_t strlen (const char *s);
 参数分析:
 s --> 需要计算长度的字符串
 返回值:
 成功 返回字符串的字符数， 不包括结束字符"\0"
```

strtok

```
strtok ( 分割字符串 )
 头文件:
 #include <string.h>
 定义函数:
 char * strtok(char *s, const char *delim);
 参数分析:
 s --> 需要分割的字符串 （需要可读写的内存地址）
 delim --> 分隔符（可以有多个分隔符）
 返回值:
  返回下一个分割后的字符串指针，
  如果已无从分割则返回 NULL.
```

注意：

strtok 第一次使用需要传递指针p， 如果想要从当前位置继续进行分割则指针p 必须写NULL

strchr

```
strchr ( 查找字符串中第一个出现的指定字符 )
strrchr // R
 头文件：
 #include <string.h>
 定义函数：
 char * strchr (const char *s, int c); // 从左往右找
 char * strrchr(const char *s, int c); // 从右往左找
 参数分析：
 s --> 需要遍历寻找的字符串 （只读地址即可）
  c --> 需要查找的字符 （虽然是个整型，但是实质是是一个无符号的字符类型）
  返回值：
  指定的字符则返回该字符所在地址，
  否则返回 0.
```

strcpy

```
strcpy ( 拷贝字符串)
strncpy 【 推荐使用 】

 头文件:
 #include <string.h>
 定义函数:
 char *strcpy(char *dest, const char *src); // 没有控制拷贝长度，有可能会溢出
 char * strncpy(char *dest, const char *src, size_t n); // 需要填写拷贝长度， 可以减少溢出
 参数分析：
  dest --> 拷贝字符串的目标地址 （可读写的内存）
  src --> 需要拷贝的字符串 （只读即可）
  n --> 控制需要拷贝的长度
  返回值：
  成功 参数 dest 的字符串起始地址

```

strcmp

```
strcmp ( 比较字符串 )
 头文件:
 #include <string.h>
 定义函数:
 int strcmp(const char *s1, const char *s2);
 int strncmp(const char *s1, const char *s2, size_t n);
 参数分析:
 s1 --> 需要比较的字符串1
 s2 --> 需要比较的字符串2
  n --> 需要比较的前 N 个字符
  返回值:
  成功 则返回0 表示两个字符串完全相同
  失败 则返回 第一给不同字符的差值

```

strcat

```
strcat ( 连接两字符串)
 头文件:
 #include <string.h>
 定义函数:
 char *strcat(char *dest, const char *src);
 char * strncat(char *dest, const char *src, size_t n);
 参数分析:
 dest --> 目标地址， 拷贝到该字符串后面
 src --> 源地址， 需要拷贝的内容
  n --> 需要拷贝的字符数
  返回值:
  返回参数 dest 的字符串起始地址
```

sprintf

```
sprintf ( 格式化字符串复制 )
 头文件:
 #include <stdio.h>
 定义函数:
```

```
int sprintf(char *str, const char * format, ...);
参数分析:
str --> 复制后的字符串（目标地址， 必须可读写）
format --> 格式化控制参数（参考打印函数）
... --> 可变参数
 返回值:
 成功则返回参数 str 字符串长度,
 失败则返回-1, 错误原因存于 errno 中.
```

bzero

```
bzero ( 将一段内存内容全清为零 )
 头文件:
 #include <string.h>
 定义函数:
 void bzero(void *s, int n);
 参数分析:
 s --> 需要清空的内存的入口地址
 n --> 需要清空的字节数
 返回值:
  无
```

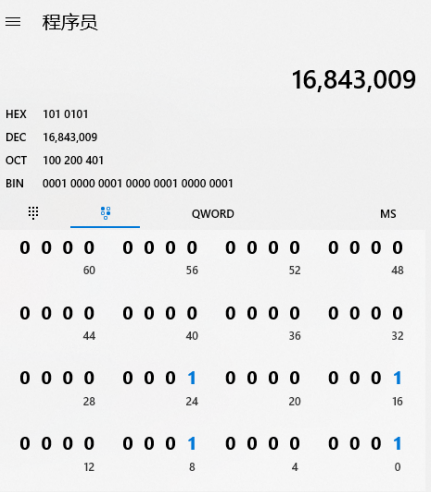
memset

```
memset ( 将一段内存空间填入某值 )
 头文件:
 #include <string.h>
 定义函数:
 void * memset(void *s, int c, size_t n);
 参数分析:
 s --> 需要设置的内存的入口地址 （可读写）
 c --> 需要填入的字符的ASCII值
 n --> 需要填入的字节数
 返回值:
 返回指向 s 的指针.
```

注意:

该函数是按字节进行写入指定值， 因此注意应该按字节来访问。

这个函数不是很适合用来清空内存， 推荐使用bzero



memcpy

```
memcpy ( 拷贝内存内容)
 头文件:
 #include <string.h>
 定义函数:
 void * memcpy (void * dest, const void *src, size_t n);
 void *memccpy(void *dest, const void *src, int c, size_t n);
 参数分析:
 dest --> 目标地址 (可读写的内存地址)
 src --> 源数据地址
  c -->
  n --> 需要拷贝的字节数
  返回值:
  返回指向 dest 的指针
```

示例:

```
char * s1 = "Hello GZ2123" ;
    char * s2 = calloc(128 , 1 ) ;
    memset(s2 , 'A', 128 );


    // strlen 计算的长度并不包括结束符， 因此应该+1让函数把结束符一起拷贝
    memcpy(s2 , s1 , strlen(s1)+1);

    printf("%s\n" , s2) ;
```

```
10
```

注意:

与strcpy 不同的地方在于，strcpy在遇到结束符时停止拷贝， 而memcpy 则是会完整拷贝内存中的前N字节，不会因为遇到结束符而停止。

memccpy 在复制的过程中会顺便检查是否出现了 字符 c， 如果出现， 则停止拷贝（拷贝C之后），后面的内容就没有继续拷贝。

《拷贝到字符C为止， 最多拷贝n个字节》

memcmp

```
memcmp ( 比较内存内容 )
 头文件:
 #include <string.h>
 定义函数:
 int memcmp (const void *s1, const void *s2, size_t n);
 参数分析:
 s1 --> 字符串1
 s2 --> 字符串2
 n --> 需要比较的前N字节
  返回值:
  成功 返回0 表示两个内容一致
  失败 返回之间的差值
```

errno

概念： 属于一个全局变量， 不需要我们自己定义。但是需要包含一个头文件

```
#include <errno.h>
```

```
strerror ( 返回错误原因的描述字符串)
 头文件:
 #include <string.h>
 定义函数:
 char * strerror(int errnum);
 参数分析:
 errnum --> 错误号码
 返回值:
```

```
返回描述错误原因的字符串指针
```

```
perror ( 打印出错误原因信息字符串 )
 头文件:
 #include <stdio.h>
 定义函数:
 void perror(const char *s);
 参数:
 s --> 用户自定义的错误提示
 返回值:
 无
```

```
for (size_t i = 0; i < 134; i++)
{
    printf("errno:%d:msg:%s\n" , i , strerror(i)); // 遍历输出所有的错误号码对应的信息
}

errno = 3 ; // 手动修改错误号码的值
perror("左勾拳失败"); // 根据错误号码之直接输出信息
```

```
errno:0:msg:Success
errno:1:msg:Operation not permitted
errno:2:msg:No such file or directory
errno:3:msg:No such process
errno:4:msg:Interrupted system call
errno:5:msg:Input/output error
errno:6:msg:No such device or address
errno:7:msg:Argument list too long
errno:8:msg:Exec format error
 errno:9:msg:Bad file descriptor
 errno:10:msg:No child processes
 errno:11:msg:Resource temporarily unavailable
 errno:12:msg:Cannot allocate memory
 errno:13:msg:Permission denied
 errno:14:msg:Bad address
 errno:15:msg:Block device required
 errno:16:msg:Device or resource busy
```

```
errno:17:msg:File exists
errno:18:msg:Invalid cross-device link
errno:19:msg:No such device
errno:20:msg:Not a directory
errno:21:msg:Is a directory
errno:22:msg:Invalid argument
errno:23:msg:Too many open files in system
errno:24:msg:Too many open files
errno:25:msg:Inappropriate ioctl for device
errno:26:msg:Text file busy
errno:27:msg:File too large
errno:28:msg:No space left on device
errno:29:msg:Illegal seek
errno:30:msg:Read-only file system
errno:31:msg:Too many links
errno:32:msg:Broken pipe
errno:33:msg:Numerical argument out of domain
errno:34:msg:Numerical result out of range
errno:35:msg:Resource deadlock avoided
errno:36:msg:File name too long
errno:37:msg:No locks available
errno:38:msg:Function not implemented
errno:39:msg:Directory not empty
errno:40:msg:Too many levels of symbolic links
errno:41:msg:Unknown error 41
errno:42:msg:No message of desired type
errno:43:msg:Identifier removed
errno:44:msg:Channel number out of range
errno:45:msg:Level 2 not synchronized
errno:46:msg:Level 3 halted
errno:47:msg:Level 3 reset
errno:48:msg:Link number out of range
errno:49:msg:Protocol driver not attached
errno:50:msg:No CSI structure available
errno:51:msg:Level 2 halted
errno:52:msg:Invalid exchange
errno:53:msg:Invalid request descriptor
errno:54:msg:Exchange full
errno:55:msg:No anode
errno:56:msg:Invalid request code
```

```
errno:57:msg:Invalid slot
errno:58:msg:Unknown error 58
errno:59:msg:Bad font file format
errno:60:msg:Device not a stream
errno:61:msg:No data available
errno:62:msg:Timer expired
errno:63:msg:Out of streams resources
errno:64:msg:Machine is not on the network
errno:65:msg:Package not installed
errno:66:msg:Object is remote
errno:67:msg:Link has been severed
errno:68:msg:Advertise error
errno:69:msg:Srmount error
errno:70:msg:Communication error on send
errno:71:msg:Protocol error
errno:72:msg:Multihop attempted
errno:73:msg:RFS specific error
errno:74:msg:Bad message
errno:75:msg:Value too large for defined data type
errno:76:msg:Name not unique on network
errno:77:msg:File descriptor in bad state
errno:78:msg:Remote address changed
errno:79:msg:Can not access a needed shared library
errno:80:msg:Accessing a corrupted shared library
errno:81:msg:.lib section in a.out corrupted
errno:82:msg:Attempting to link in too many shared libraries
errno:83:msg:Cannot exec a shared library directly
errno:84:msg:Invalid or incomplete multibyte or wide character
errno:85:msg:Interrupted system call should be restarted
errno:86:msg:Streams pipe error
errno:87:msg:Too many users
errno:88:msg:Socket operation on non-socket
errno:89:msg:Destination address required
errno:90:msg:Message too long
errno:91:msg:Protocol wrong type for socket
errno:92:msg:Protocol not available
errno:93:msg:Protocol not supported
errno:94:msg:Socket type not supported
errno:95:msg:Operation not supported
errno:96:msg:Protocol family not supported
```

```
errno:97:msg:Address family not supported by protocol
errno:98:msg:Address already in use
errno:99:msg:Cannot assign requested address
errno:100:msg:Network is down
errno:101:msg:Network is unreachable
errno:102:msg:Network dropped connection on reset
errno:103:msg:Software caused connection abort
errno:104:msg:Connection reset by peer
errno:105:msg:No buffer space available
errno:106:msg:Transport endpoint is already connected
errno:107:msg:Transport endpoint is not connected
errno:108:msg:Cannot send after transport endpoint shutdown
errno:109:msg:Too many references: cannot splice
errno:110:msg:Connection timed out
errno:111:msg:Connection refused
errno:112:msg:Host is down
errno:113:msg:No route to host
errno:114:msg:Operation already in progress
errno:115:msg:Operation now in progress
errno:116:msg:Stale file handle
errno:117:msg:Structure needs cleaning
errno:118:msg:Not a XENIX named type file
errno:119:msg:No XENIX semaphores available
errno:120:msg:Is a named type file
errno:121:msg:Remote I/O error
errno:122:msg:Disk quota exceeded
errno:123:msg:No medium found
errno:124:msg:Wrong medium type
errno:125:msg:Operation canceled
errno:126:msg:Required key not available
errno:127:msg:Key has expired
errno:128:msg:Key has been revoked
errno:129:msg:Key was rejected by service
errno:130:msg:Owner died
errno:131:msg:State not recoverable
errno:132:msg:Operation not possible due to RF-kill
errno:133:msg:Memory page has hardware error

errno:134:msg:Unknown error 134
```

练习：

1. 实现一下课堂中所写的demo （自己手动输入一个字符串， 通过strtok 来分割）
2. 尝试把青云客的返回内容进行分割
3. 尝试使用strstr来寻找需要的信息

类似如下：

```
广州天气 ：当前温度24℃，感冒低发期，天气舒适，请注意多吃蔬菜水果，多喝水哦。
[04月15日]：大雨，低温 20℃，高温 27℃，风力3级
[04月16日]：中雨，低温 20℃，高温 23℃，风力2级
[04月17日]：雷阵雨，低温 19℃，高温 25℃，风力3级
[04月18日]：多云，低温 18℃，高温 24℃，风力2级
[04月19日]：多云，低温 18℃，高温 24℃，风力1级
```

作业：

尝试自己写一个分割字符串的函数，要求分割符是一个组合的整体 "{br}"

编写一个函数，接收一个给定的字符串，返回去除所有的空格后的字符串。

编写一个程序，将两个给定的字符串连接起来，要求不能用 strcat 或 strncat 函数。

完成课堂练习中未完成部分。

预习：

作用域

存储期

结构体