知识点1【atoi atol atof 】将字符串转化为数值

头文件：stdlib.h

```c
#include<stdio.h>
#include<stdlib.h>//atoi atol
void test01()
{
 printf("%d\n",atoi("123"));
 printf("%ld\n",atol("12345"));
 printf("%f\n",atof("12.3"));

 return;
}
int main(int argc,char *argv[])
{
 test01();
 return 0;
}
```

运行结果：

```
123
12345
12.300000
Press any key
```

自定义函数实现atoi的功能：

```
int my_atoi(char *buf)
{
 //buf="1234"
 int sum = 0;
 while(*buf != '\0' && (*buf>='0' && *buf<='9'))
 {
 sum = sum*10+(*buf++)-'0';
 }

 return sum;
}
void test02()
{
 char buf[128]="";
 printf("请输入字符串:");
 scanf("%s",buf);

 printf("%d\n",my_atoi(buf));
 return;
}
int main(int argc,char *argv[])
{
 test02();
 return 0;
}
```

运行结果：

```
C:\WORK\C\CODE\DAY11\00_test\Debug\00_test.exe
请输入字符串:123abc
123
Press any key to continue
```

知识点2【字符串切割函数 strtok】(了解)

```
char *strtok(char *str, const char *delim);
第一次切割： str必须指向要切割的字符串首元素地址 delim指向要切割的符号
第2~n次切割：str必须指向NULL delim指向要切割的符号

返回值：切割成功 返回切割到字符换片段的首元素地址 失败：NULL

注意：strtok不能切割字符串常量
```

```
"hehe:haha:xixi:lala"
如果切割完成会产生 "hehe" "haha" "xixi" "lala"
一般选择 char *arr[] 指针数组 来存放上面 独立的字符串的首元素地址
```

案例1：

```
void test03()
{
 char buf[]="hehehe:haha:xixi:lala:heihei:henhen";
 char *arr[32]={NULL};
 int i=0;

 //第1次切割
 arr[i] = strtok(buf, ":");

 //第2~n切割
 while(arr[i] != NULL)//保证上一次切割正常 才有进行下一次切割的必要
 {
 i++;
 arr[i] = strtok(NULL,":");
 }

 //遍历切割的内容
 i=0;
```

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

}
```

运行结果：

```
hehehe
haha
xixi
lala
heihei
henhen
Press any key to continue
```

案例：

```
void test04()
{
 char buf[]="hehehe:haha:xixi:lala:heihei:henhen";
 char *arr[32]={buf,NULL};
 int i=0;

 //切割
 while(1)
 {
 arr[i] = strtok(arr[i],":");
 if(arr[i] == NULL)
 break;
 i++;
 }

 //遍历切割的内容
 i=0;
 while(arr[i] != NULL)
```

```
 {
 printf("%s\n",arr[i]);
 i++;
 }

}
```

案例：

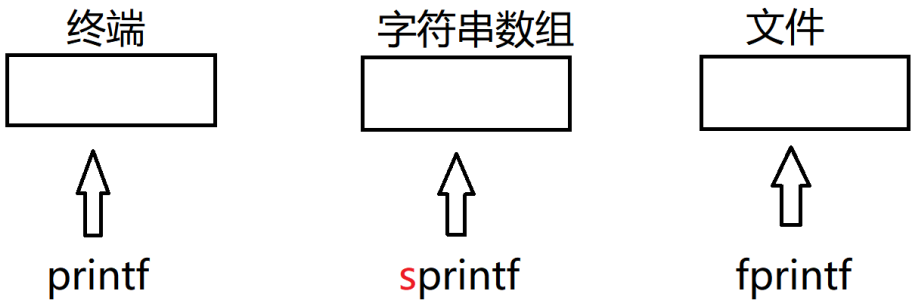
```
void test05()
{
 char buf[]="hehehe:haha:xixi:lala:heihei:henhen";
 char *arr[32]={buf,NULL};
 int i=0;

 //切割
 while(arr[i++] = strtok(arr[i],":"));

 //遍历切割的内容
 i=0;
 while(arr[i] != NULL)
 {
 printf("%s\n",arr[i]);
 i++;
 }

}
```

注意：

```
void test05()
{
 char buf[]="hehehe:haha#xixi@lala:::::::heihei####henhen";
 char *arr[32]={buf,NULL};
 int i=0;

 //切割
 while(arr[i++] = strtok(arr[i],":#@"));

 //遍历切割的内容
 i=0;
 while(arr[i] != NULL)
 {
```

```
 printf("%s\n",arr[i]);
 i++;
 }

}
```

知识点3【sprintf组包】（了解）



案例：

```
int sprintf(buf,"格式",数据)
//buf:用来存放组好的报文
//"格式"：按照格式组包
//数据：各个零散的数据

返回值：返回值的是组好的报文的实际长度（不包含'\0'）
```

```
void test06()
{
 int year = 2020;
 int mon = 2;
 int day = 24;
 int len = 0;

 //需求：将2020 2 24组成一个"2020年2月24日"
 char buf[128]="";

 //%d 和 year 类型要一一对应
 len = sprintf(buf,"%d年%d月%d日", year,mon,day);
 printf("len = %d\n", len);
 printf("buf=%s\n",buf);
}
```

运行结果：

```
len = 13
buf=2020年2月24日
Press any key to continue
```

案例:

```
1  void test07()
2  {
3   char name[]="梁何";
4   int age = 18;
5   char sex[]="男";
6   char addr[]="重庆市潼南区";
7
8   char buf[128]="";
9   int len = 0;
10   len = sprintf(buf,"姓名:%s 年龄:%d 性别:%s 地址:%s",name,age,sex, addr);
11   printf("len = %d\n",len);
12   printf("buf=%s\n",buf);
13
14  }
```

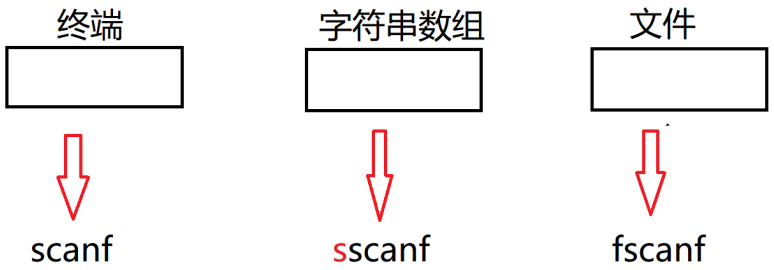
运行结果:

```
len = 43
buf=姓名:梁何 年龄:18 性别:男 地址:重庆市潼南区
Press any key to continue
```

将数值转成字符串

```
1  void test08()
2  {
3   int num = 2020;
4   char buf[128]="";
5
6   sprintf(buf,"%d",num);
7   printf("buf=%s\n", buf);//"2020"
8  }
```

知识点4【sscanf解包】



案例1：sscanf的基本语法

```
void test09()
{
 char buf[]="2020年2月24日";
 int year=0;
 int mon=0;
 int day=0;
 char ch = 0;
 char str_year[32]="";
 char str_mon[32]="";
  char str_day[32]="";


 //%d 只能提取'0'~'9'
 sscanf(buf,"%d年%d月%d日", &year,&mon,&day );
 printf("year = %d\n",year);//2020
 printf("mon = %d\n",mon);//2
 printf("day = %d\n",day);//24

 //%c 提取一个字符 %f 提取提取浮点数
 sscanf(buf,"%c", &ch);
 printf("##%c##\n",ch);//'2'

 //%s 提取一个字符串 遇到空格、回车、'\0' 停止获取
 //buf==>"2020年2月24日"
 sscanf(buf,"%s年%s月%s日",str_year, str_mon,str_day );
 printf("str_year=%s\n",str_year);//"2020年2月24日"
 printf("str_mon=%s\n",str_mon);//null
 printf("str_day=%s\n",str_day);//null
}
```

运行结果：

```
year = 2020
mon = 2
day = 24
##2##
str_year=2020年2月24日
str_mon=
str_day=
Press any key to continue
```

sscanf的高级用法1：使用%*s %*d 跳过提取的内容（不要提取的内容）

```c
void test10()
{
 int data1=0;
 //sscanf("1234 5678","1234 %d",&data1);//5678
 //sscanf("1234 5678","%*d %d",&data1);//5678
 sscanf("1234 5678","%*s %d",&data1);//5678
 printf("data1=%d\n",data1);
}
```

sscanf的高级用法2：使用%[n]s %[n]d 提取指定宽度n的字符串或数据

```c
void test11()
{
 int data1 = 0;
 int data2 = 0;
 char buf[16]="";

 sscanf("12abc5678","%*5s%d" ,&data1);//5678
 printf("data1=%d\n",data1);


  sscanf("12345678","%*2s%2d%*2d%s" ,&data2, buf);//data2=34 buf="78";
  printf("data2 = %d\n",data2);//34
  printf("buf=%s\n", buf);//"78"
 }
```

运行结果：

```
data1=5678
data2 = 34
buf=78
Press any key to continue
```