标准I/O（六）

主讲：大海老师

课程目标

sprintf （熟练）

fprintf（熟练）

小结

标准I/O – 格式化输出

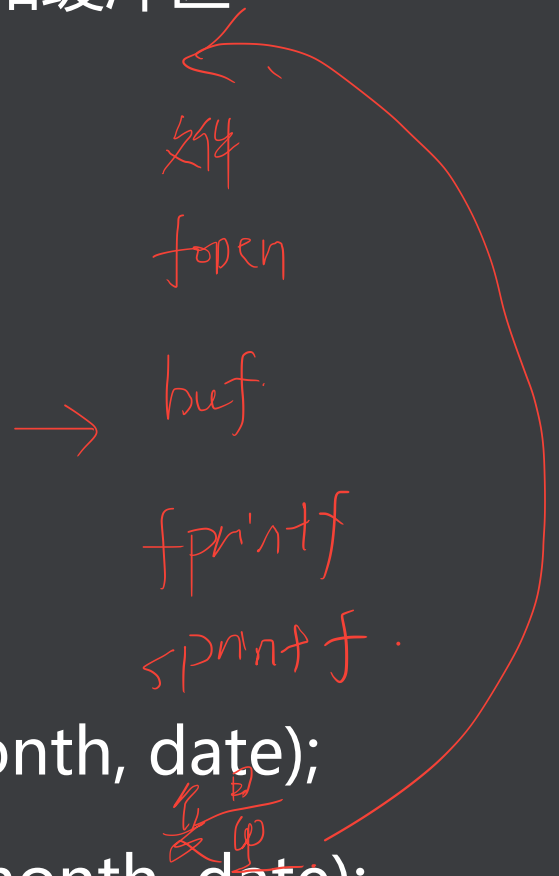
```
#include  <stdio.h>

int printf(const char *fmt, ...);

int fprintf(FILE *stream, const char *fmt, ...);

int sprintf(char *s, const char *fmt, ...);
```

- 成功时返回输出的字符个数；出错时返回EOF
- 使用起来很方便，强烈推荐！

标准I/O – 格式化输出 – 示例

以指定格式 “年-月-日” 分别写入文件和缓冲区

```
int  year, month, date;
FILE *fp;
char buf[64];
year = 2014; month = 10; date = 26;
fp = fopen( “test.txt” , “a+” );
fprintf(fp, “%d-%d-%d\n” , year, month, date);
sprintf(buf, “%d-%d-%d\n” , year, month, date);
```



标准I/O – 格式化输入

- int fscanf(FILE *stream, const char *format, ...);
- int sscanf(const char *str, const char *format, ...);

标准I/O – 小结

fprintf

sprintf

fscanf

sscanf

注：对于字符数组 strlen是实际大小，sizeof是总大小

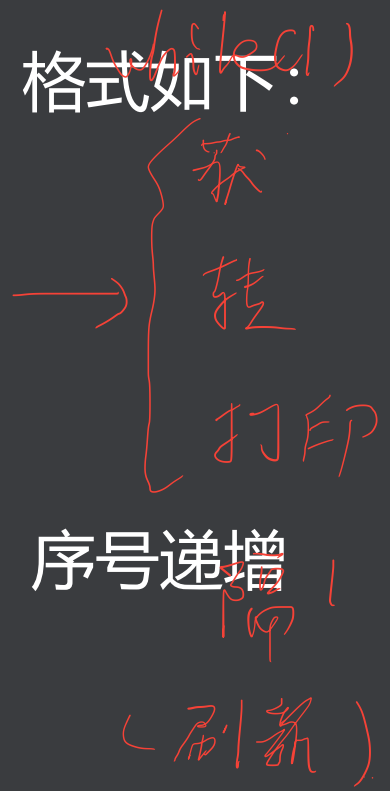
标准I/O – 思考和练习

每隔1秒向文件test.txt中写入当前系统时间，格式如下：

1，2014-10-15 15:16:42
2，2014-10-15 15:16:43

该程序无限循环，直到按Ctrl-C中断程序

每次执行程序时，系统时间追加到文件末尾，序号递增



1，2014-10-15 15:16:42
2，2014-10-15 15:16:43
3，2014-10-16 11:35:07
4，2014-10-16 11:35:08

标准I/O – 思考和练习– 提示

time()用来获取系统时间(秒数)

time_t time(time_t *seconds) 1970.1.1 0:0:0

localtime()将系统时间转换成本地时间

struct tm *localtime(const time_t *timer)

```
struct tm {
   int tm_sec;        /* 秒，范围从 0 到 59              */
   int tm_min;        /* 分，范围从 0 到 59              */
   int tm_hour;       /* 小时，范围从 0 到 23              */
   int tm_mday;       /* 一月中的第几天，范围从 1 到 31                */
   int tm_mon;        /* 月份，范围从 0 到 11              */
   int tm_year;       /* 自 1900 起的年数                */
   int tm_wday;       /* 一周中的第几天，范围从 0 到 6             */
   int tm_yday;       /* 一年中的第几天，范围从 0 到 365                 */
   int tm_isdst;      /* 夏令时                        */
};
```

标准I/O – 思考和练习 – 提示

sleep()实现程序睡眠

以何种方式打开流?

流的缓冲类型对文件写入的影响