练习：实现"head -n 文件名"功能.

思路：打开文件，循环读文件，当读到文件末尾时循环结束，循环中计算当前读的行数，行数和 n 进行比较，相等时循环结束。

./a.out -5 file.c -> argv[1]:"-5" "5"->5

将字符串转换成整数：int num = atoi(“123”); //num:123

标准 IO

1. 函数接口

1.1 二进制读写

```
size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream);
功能：从文件流读取多个元素
参数：  ptr ：用来存放读取元素
        size ：元素大小  sizeof(数据类型)
        nmemb ：读取对象的个数
        stream ：要读取的文件
返回值：成功：读取对象的个数
      读到文件尾或失败：0
size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE
*stream);
功能：按对象写
参数：同上
返回值：成功：写的元素个数
      失败 ：-1
Fread 和 fwrite 函数注意：
1）两个函数的返回值为：读或写的对象数
2）对于二进制数据我们更愿意一次读或写整个结构。
```

1.2 文件定位操作

```
void rewind(FILE *stream);
功能：将文件位置指针定位到起始位置
int fseek(FILE *stream, long offset, int whence);
功能：文件的定位操作
参数：stream：文件流
      offset：偏移量：正数表示向后文件尾部偏移，负数表示向文件开头偏移
      whence：相对位置：
            SEEK_SET:相对于文件开头
            SEEK_CUR:相对于文件当前位置
            SEEK_END:相对于文件末尾
返回值：成功：0
         失败：-1
注：当打开文件的方式为 a 或 a+时，fseek 不起作用
long ftell(FILE *stream);
功能：获取当前的文件位置
参数：要检测的文件流
返回值：成功：当前的文件位置，出错：-1
```

1.3 重定向打开文件

```
FILE * freopen(const char *pathname,  const char *mode,  FILE*
fp)
功能：将指定的文件流重定向到打开的文件中
参数：path：文件路径
mode：打开文件的方式（同 fopen）
        fp：文件流指针
返回值：成功：返回文件流指针
        失败：NULL
```

百度+man 手册

time(time_t *tm)

函数调用：

参数：个数、类型和函数原型意义对应；当函数原型中参数是一级指针时，需要定义变量传地址

返回值：并不是所有函数都需要接收返回值；如果需要接收返回值，函数原型返回值类型是什么，在代码定义什么类型变量或指针去接收

文件 IO

1. 概念

1.1 定义

在 posix(可移植操作系统接口)中定义的一组输入输出的函数

系统调用：内核向上提供的一组接口

1.2 特点

1）没有缓冲机制，每次 IO 操作都会引起系统调用

2）围绕文件描述符操作，非负整数(int)，依次分配

3）默认打开三个文件描述符：0(标准输入)、1(标准输出)、2(标准错误)

4）可以操作除 d 以外其他任意类型文件

2. 函数接口

2.1 打开文件

```
int open(const char *pathname, int flags);
功能：打开文件
参数：pathname：文件路径名
      flags：打开文件的方式
            O_RDONLY：只读
            O_WRONLY:只写
            O_RDWR：可读可写
            O_CREAT:创建
            O_TRUNC：清空
            O_APPEND：追加
返回值：成功：文件描述符
        失败：-1
当第二个参数中有 O_CREAT 选项时，需要给 open 函数传递第三个参数，指定创建文件的权限
int open(const char *pathname, int flags, mode_t mode);
创建出来的文件权限为指定权限值&(~umask)  //umask 为文件权限掩码
```

打开文件方式标准 IO 和文件 IO 对应关系：

标准 IO	文件 IO
r	O_RDONLY
r+	O_RDWR
w	O_WRONLY|O_CREAT|O_TRUNC,0666
w+	O_RDWR|O_CREAT|O_TRUNC,0666
a	O_WRONLY|O_CREAT|O_APPEND,0666
a+	O_RDWR|O_CREAT|O_APPEND,0666

2.2 读写文件

```
ssize_t read(int fd, void *buf, size_t count);
功能：从一个已打开的可读文件中读取数据
参数：fd   文件描述符
      buf   存放位置
     count   期望的个数
返回值：成功：实际读到的个数
        返回-1：表示出错,并设置 errno 号
        返回 0：表示读到文件结尾
ssize_t write(int fd, const void *buf, size_t count);
功能：向指定文件描述符中，写入 count 个字节的数据。
参数：fd    文件描述符
           buf    要写的内容
           count   期望值
返回值：成功：实际写入数据的个数
                失败  :  -1
```

练习：实现 cp 功能。

cp srcfile newfile -> ./a.out srcfile newfile

思路：打开两个文件，循环读源文件写新文件，当读到源文件末尾时循环结束

diff 文件名 1 文件名 2：比较两个文件是否相等

2.3 关闭文件

```
int close(int fd);
参数：fd：文件描述符
```

2.4 文件定位

```
off_t lseek(int fd, off_t offset, int whence);
功能：设定文件的偏移位置
参数：fd：文件描述符
    offset 偏移量
        正数：向文件结尾位置移动
        负数：向文件开始位置
    whence  相对位置
        SEEK_SET    开始位置
        SEEK_CUR    当前位置
        SEEK_END    结尾位置
返回值：成功：文件的当前位置
```

作业：

1. 梳理今天学习内容
2. 通过标准 IO 实现 cp 功能
3. 实现如下功能
   1-- 打开一个文件，不存在创建，存在清零
   2-- 向文件中第 10 位置处写一个字符，
   3-- 在文件此时的位置，后 20 个位置处，写一行字符串 hello 进去
   4-- 求文件的长度。