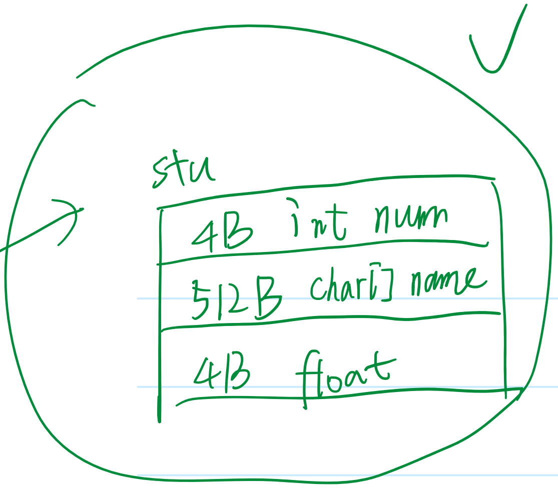
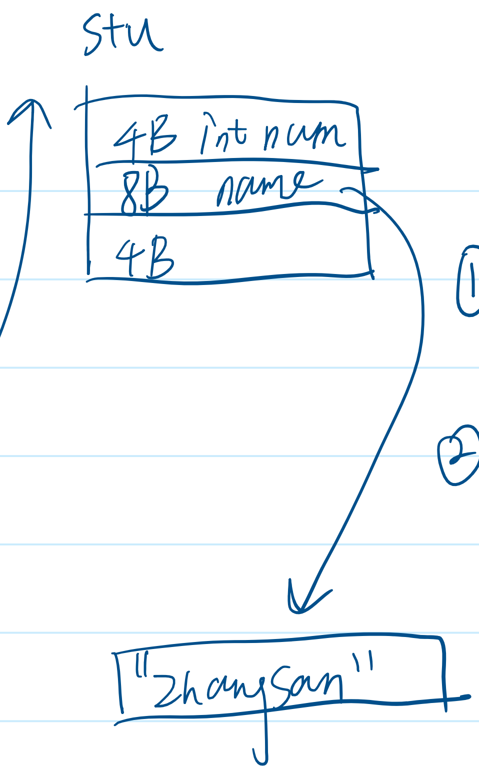
结构体设计

2023年8月5日 10:20

```c
typedef struct student_s {
    int num;
    // char name[512];
    // char *name;
    float score;
} student_t;
int main(int argc, char *argv[])
{
    student_t stu;
    return 0;
}
```



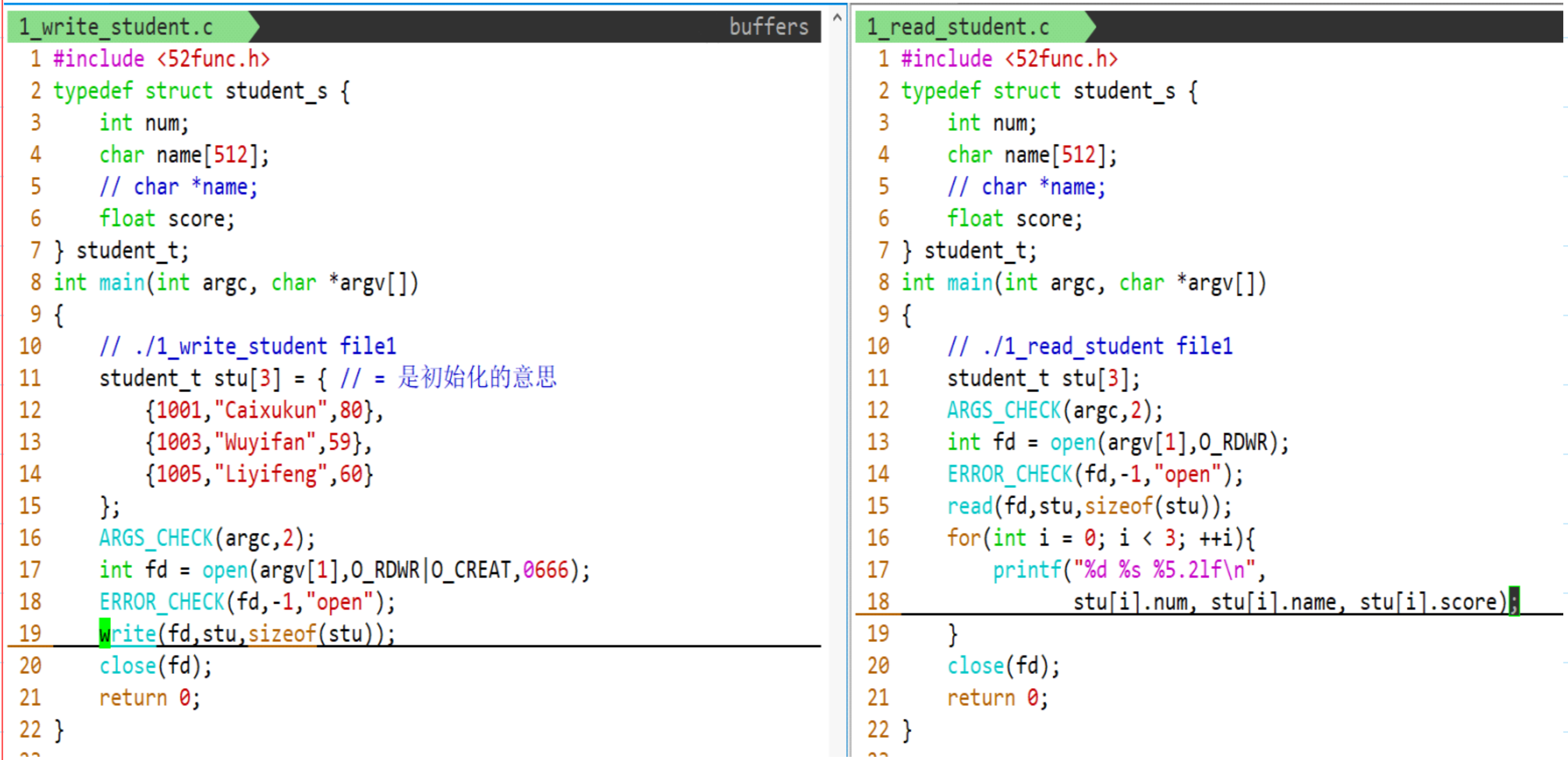
json/protobuf



① name字段外申请内存.

② 不能写入文件.

2023年8月5日　　10:36



1_write_student.c　　buffers

```c
#include <52func.h>
typedef struct student_s {
    int num;
    char name[512];
    // char *name;
    float score;
} student_t;
int main(int argc, char *argv[])
{
    // ./1_write_student file1
    student_t stu[3] = { // = 是初始化的意思
        {1001,"Caixukun",80},
        {1003,"Wuyifan",59},
        {1005,"Liyifeng",60}
    };
    ARGS_CHECK(argc,2);
    int fd = open(argv[1],O_RDWR|O_CREAT,0666);
    ERROR_CHECK(fd,-1,"open");
    write(fd,stu,sizeof(stu));
    close(fd);
    return 0;
}
```

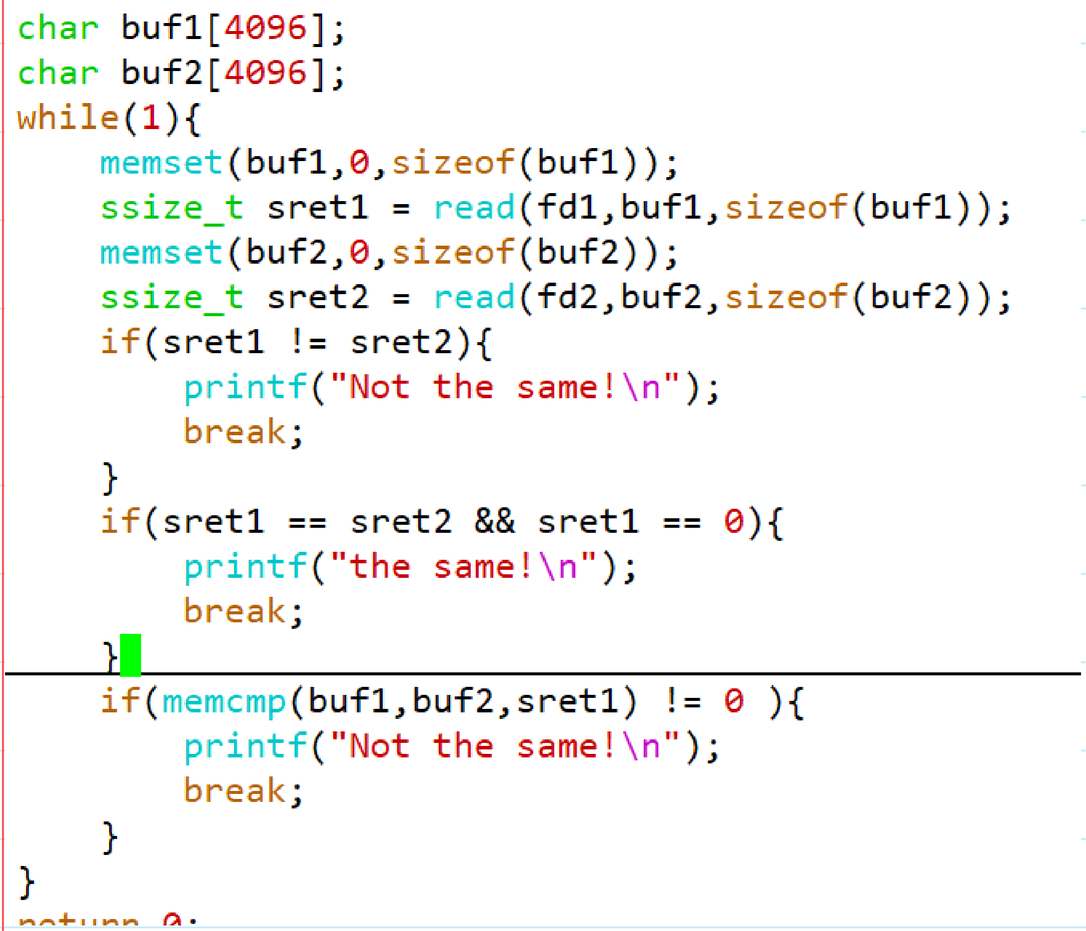
1_read_student.c

```c
#include <52func.h>
typedef struct student_s {
    int num;
    char name[512];
    // char *name;
    float score;
} student_t;
int main(int argc, char *argv[])
{
    // ./1_read_student file1
    student_t stu[3];
    ARGS_CHECK(argc,2);
    int fd = open(argv[1],O_RDWR);
    ERROR_CHECK(fd,-1,"open");
    read(fd,stu,sizeof(stu));
    for(int i = 0; i < 3; ++i){
        printf("%d %s %5.2lf\n",
               stu[i].num, stu[i].name, stu[i].score);
    }
    close(fd);
    return 0;
}
```

2023年8月5日 11:05

```c
int main(int argc, char *argv[])
{
    // ./3_1million file1
    ARGS_CHECK(argc,2);
    int fd = open(argv[1],O_RDWR);
    ERROR_CHECK(fd,-1,"open");
    //char ch = '1';
    //for(int i = 0; i < 1000000; ++i){
    //    write(fd,&ch,1);
    //}
    char buf[10000];
    memset(buf,'1',sizeof(buf));
    for(int i = 0; i < 100; ++i){
        write(fd,buf,sizeof(buf));
    }
    close(fd);
    return 0;
}
```

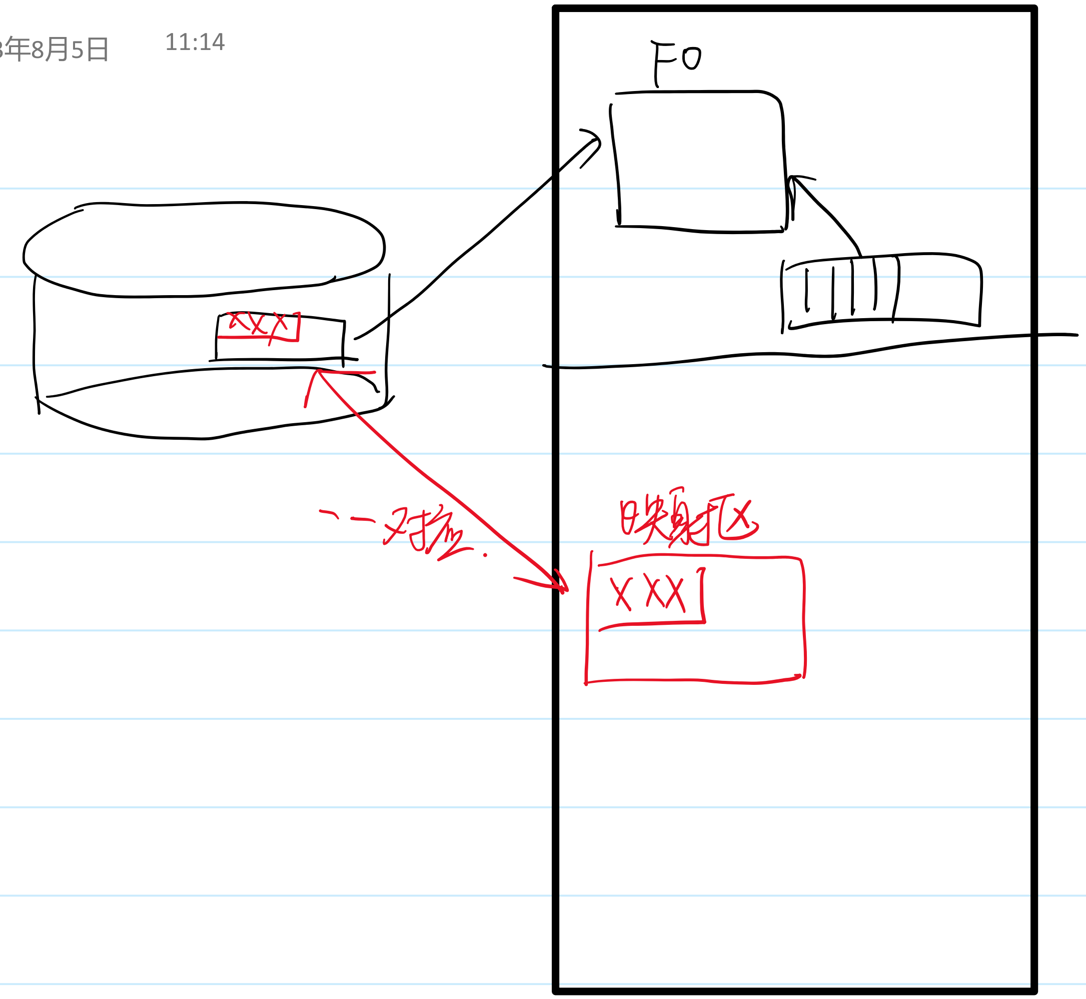
2023年8月5日 11:13



```
char buf1[4096];
char buf2[4096];
while(1){
    memset(buf1,0,sizeof(buf1));
    ssize_t sret1 = read(fd1,buf1,sizeof(buf1));
    memset(buf2,0,sizeof(buf2));
    ssize_t sret2 = read(fd2,buf2,sizeof(buf2));
    if(sret1 != sret2){
        printf("Not the same!\n");
        break;
    }
    if(sret1 == sret2 && sret1 == 0){
        printf("the same!\n");
        break;
    }
    if(memcmp(buf1,buf2,sret1) != 0 ){
        printf("Not the same!\n");
        break;
    }
}
return 0;
```

文件映射

2023年8月5日 11:14



读文件. 访问内存的方式

read/write → ✗

mmap

2023年8月5日 11:20

```
void *mmap(void *addr, size_t length, int prot, int flags,
                 int fd, off_t offset);
int munmap(void *addr, size_t length);
```

NULL mmap自动分配

① open O_RDWR

② 填入 PROT_READ|PROT_WRITE

MAP_SHARED

fd

0.

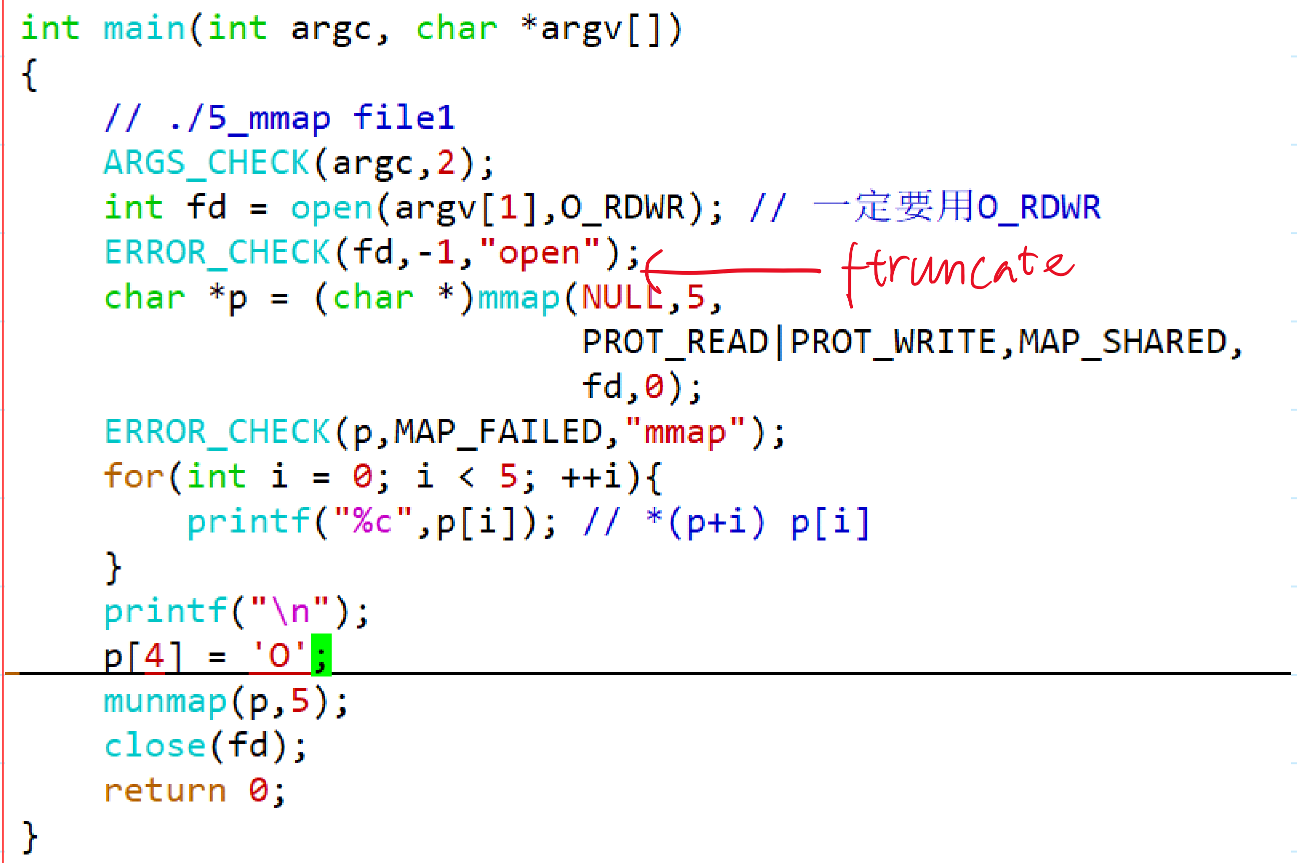
映射区长度 mmap不能改变文件大小

配合 ftruncate

映射区首地址

使用mmap之前，offset处于文件开始.

2023年8月5日 11:35



```c
int main(int argc, char *argv[])
{
    // ./5_mmap file1
    ARGS_CHECK(argc,2);
    int fd = open(argv[1],O_RDWR); // 一定要用O_RDWR
    ERROR_CHECK(fd,-1,"open");
    char *p = (char *)mmap(NULL,5,
                            PROT_READ|PROT_WRITE,MAP_SHARED,
                            fd,0);
    ERROR_CHECK(p,MAP_FAILED,"mmap");
    for(int i = 0; i < 5; ++i){
        printf("%c",p[i]); // *(p+i) p[i]
    }
    printf("\n");
    p[4] = 'O';
    munmap(p,5);
    close(fd);
    return 0;
}
```

ftruncate

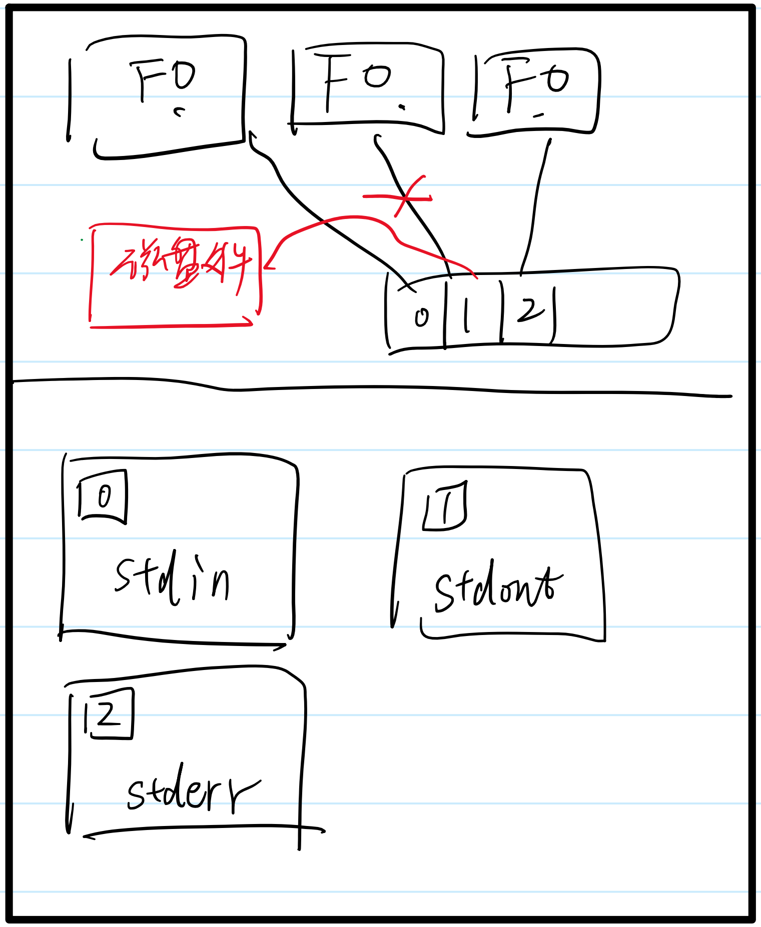
read/write 顺序

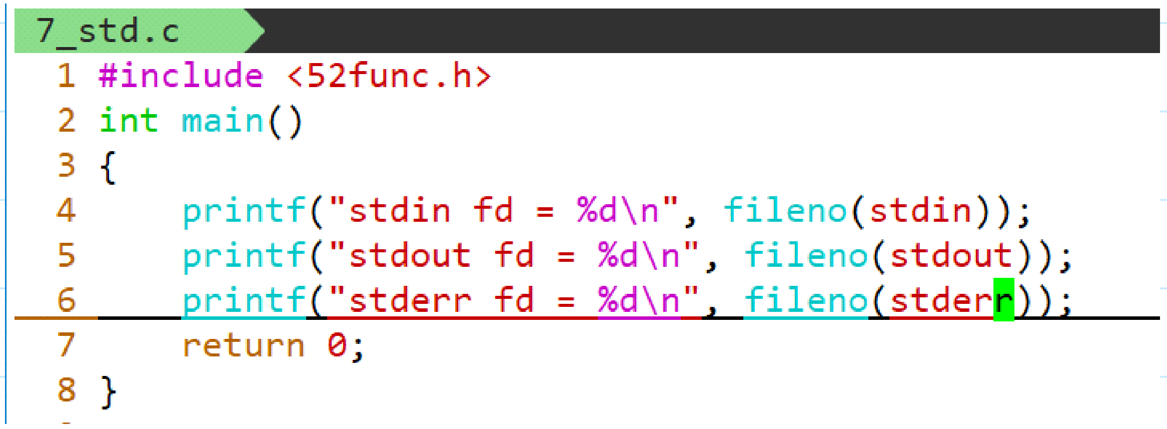
mmap 随机访问.

在程序一启动的时候会打开3个文件流

2023年8月5日 11:39

FILE 用户态文件缓冲区

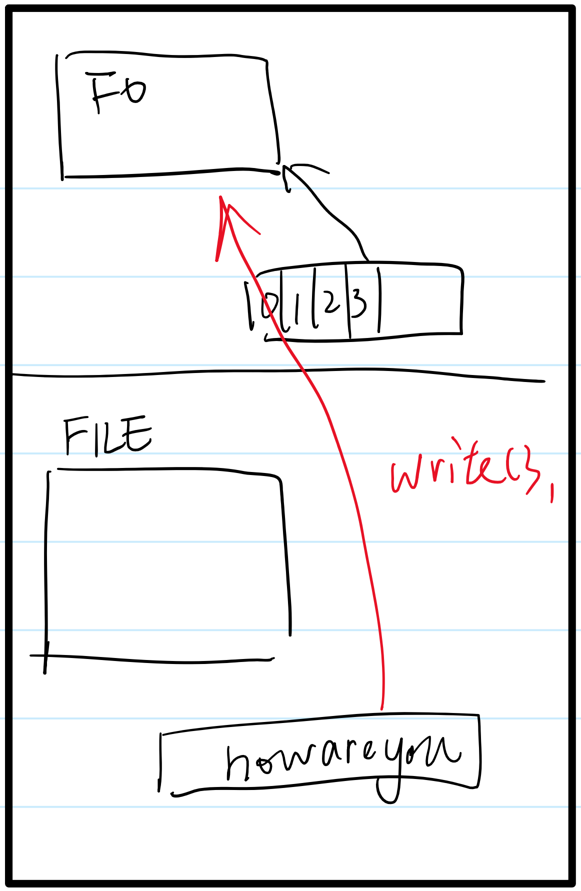



```c
7_std.c
#include <52func.h>
int main()
{
    printf("stdin fd = %d\n", fileno(stdin));
    printf("stdout fd = %d\n", fileno(stdout));
    printf("stderr fd = %d\n", fileno(stderr));
    return 0;
}
```

文件流的底层是文件对象

2023年8月5日 11:41

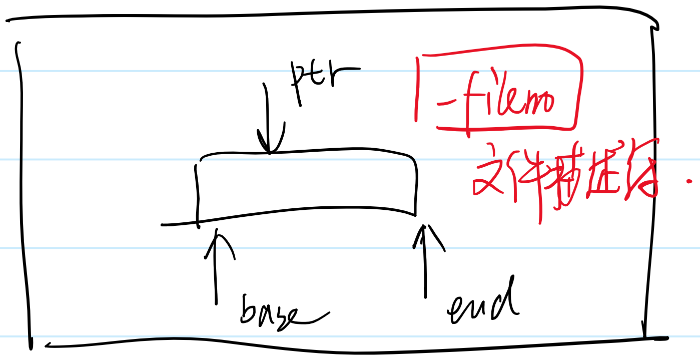
```c
int main(int argc, char *argv[])
{
    // ./6_fileno file1
    ARGS_CHECK(argc,2);
    FILE *fp = fopen(argv[1],"r+"); // 创建了文件流
    ERROR_CHECK(fp,NULL,"fopen");
    write(3,"howareyou",9); // 用文件描述符写入
    fclose(fp);
    return 0;
}
```

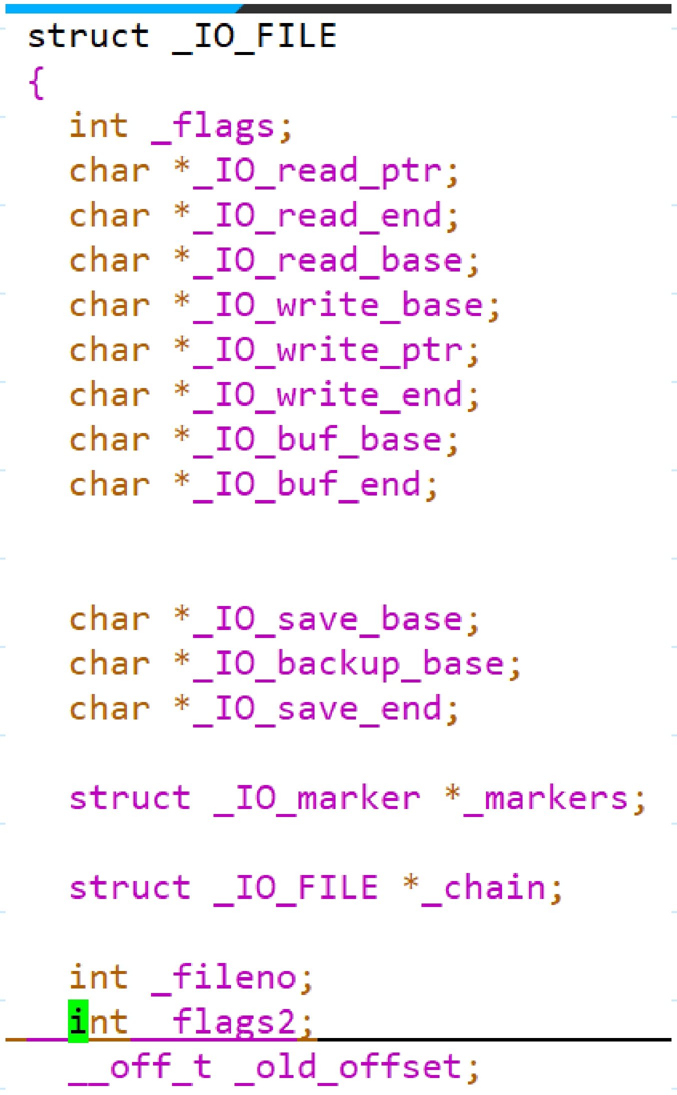


```c
// write(3,"howareyou",9); // 用文件描
write(fp->_fileno,"howoldareyou",12);
```

FILE

2023年8月5日 11:48

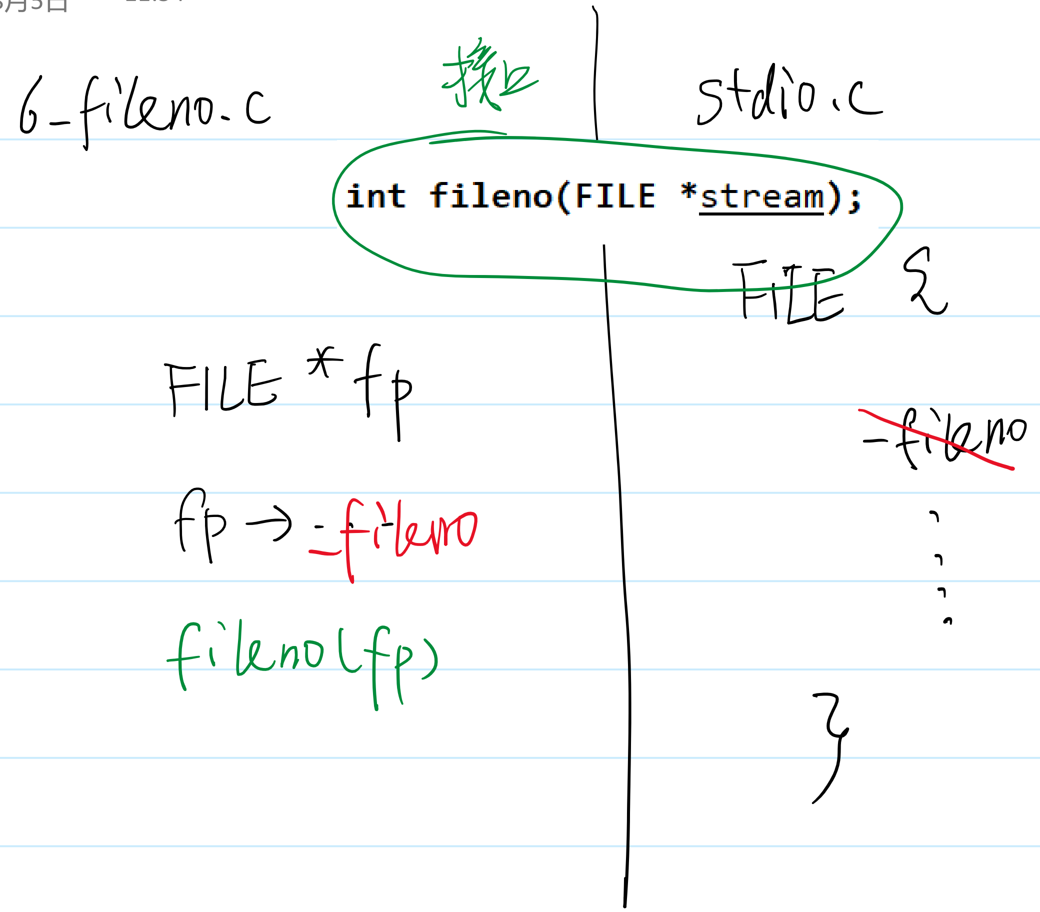




```
struct _IO_FILE
{
  int _flags;
  char *_IO_read_ptr;
  char *_IO_read_end;
  char *_IO_read_base;
  char *_IO_write_base;
  char *_IO_write_ptr;
  char *_IO_write_end;
  char *_IO_buf_base;
  char *_IO_buf_end;


  char *_IO_save_base;
  char *_IO_backup_base;
  char *_IO_save_end;

  struct _IO_marker *_markers;

  struct _IO_FILE *_chain;

  int _fileno;
  int _flags2;
  __off_t _old_offset;
```

面向接口编程

2023年8月5日 11:54



```
int fileno(FILE *stream);
```

面向接口编程，而不是面向实现编程

_fileno1

Δ

实现