1 结构体嵌套二级指针练习

2 结构体偏移量

2.1 获取属性偏移

2.1.1 offsetof

2.1.2 (int)&(p->b) - (int)p

2.2 通过偏移量 获取内存

2.3 结构体嵌套结构体

3 内存对齐

3.1 查看对齐模数 #pragma pack(*show*)

3.2 默认对齐模数 8

3.3 自定义数据类型 对齐规则

3.3.1 第一个属性开始 从 0 开始偏移

3.3.2 第二个属性开始 要放在 该类型的大小 与 对齐模数比 取小的值 的整数倍

3.3.3 所有属性都计算完后，再整体做二次偏移，将整体计算的结果 要放在 结构体最大类型 与对齐模数比 取小的值的 整数倍上

3.4 结构体嵌套结构体

3.4.1 结构体嵌套结构体时候，子结构体放在该结构体中最大类型 和对齐模数比的整数倍上即可

4 文件读写回顾

4.1 按照字符读写

4.1.1 写 fputc

4.1.2 读 fgetc

4.1.3 while ((ch = *fgetc*(f_read)) != *EOF*) 判断是否到文件尾

4.2 按行读写

4.2.1 写 fputs

4.2.2 读 fgets

4.3 按块读写

4.3.1 写 fwrite

4.3.1.1 参数 1 数据地址 参数 2 块大小 参数 3 块个数 参数 4 文件指针

4.3.2 读 fread

4.4 格式化读写

4.4.1 写 fprintf

4.4.2 读 fscanf

4.5 随机位置读写

4.5.1 fseek(文件指针， 偏移， 起始位置)

4.5.1.1 SEEK_SET 从头开始

4.5.1.2 SEEK_END 从尾开始

4.5.1.3 SEEK_CUR 从当前位置

4.5.2 rewind 将文件光标置首

4.6 error 宏 利用 perror 打印错误提示信息

5 文件读写注意事项

5.1 注意事项 1：

5.1.1 不要用 feof 按照字符方式读文件，原因有滞后性，会读出 EOF

5.2 注意事项 2：

5.2.1 如果属性开辟到堆区，不要存指针到文件中，要将指针指向的内容存放到文件中

6 配置文件读写案例

6.1 文件中按照键值对方式 存放了有效的信息需要解析出来

6.2 创建 config.h 和 config.c 做配置文件读操作

6.3 获取有效信息的行数 getFileLines

6.4 判断字符串是否是有效行 `int isValidLines(char *str)`

6.5 解析文件到配置信息数组中

6.5.1 `void parseFile(char * filePath, int lines , struct ConfigInfo ** configinfo);`

6.6 通过 key 获取 value 值

6.6.1 `char * getInfoByKey(char * key, struct ConfigInfo * configinfo, int len);`

6.7 释放内存

6.7.1 `void freeConfigInfo(struct ConfigInfo * configinfo);`

7