1. 结构体嵌套二级指针练习
2. 结构体偏移量
   1. 获取属性偏移
      1. offsetof
      2. (int)&(p->b) - (int)p
   2. 通过偏移量 获取内存
   3. 结构体嵌套结构体
3. 内存对齐
   1. 查看对齐模数 #pragma pack(*show*)
   2. 默认对齐模数 8
   3. 自定义数据类型 对齐规则
      1. 第一个属性开始 从0开始偏移
      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. 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
   4. 结构体嵌套结构体
      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
   1. 按照字符读写
      1. 写 fputc
      2. 读 fgetc
      3. while ( (ch = *fgetc*(f\_read)) != *EOF* ) 判断是否到文件尾
   2. 按行读写
      1. 写 fputs
      2. 读 fgets
   3. 按块读写
      1. 写 fwrite
         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
      2. 读 fread
   4. 格式化读写
      1. 写 fprintf
      2. 读 fscanf
   5. 随机位置读写
      1. fseek( 文件指针， 偏移， 起始位置 )
         1. SEEK\_SET 从头开始
         2. SEEK\_END 从尾开始
         3. SEEK\_CUR 从当前位置
      2. rewind 将文件光标置首
   6. error宏 利用perror打印错误提示信息
5. 文件读写注意事项
   1. 注意事项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
   2. 注意事项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
   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
   2. 创建 config.h 和 config.c做配置文件读操作
   3. 获取有效信息的行数 getFileLines
   4. 判断字符串是否是有效行 int isValidLines(char \*str)
   5. 解析文件到配置信息数组中
      1. void parseFile(char \* filePath, int lines , struct ConfigInfo \*\* configinfo);
   6. 通过key获取value值
      1. char \* getInfoByKey(char \* key, struct ConfigInfo \* configinfo, int len);
   7. 释放内存
      1. void freeConfigInfo(struct ConfigInfo \* configinfo);