1. 海滩上有一堆桃子，五只猴子来分。第一只猴子把这堆桃子平均分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成五份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第三、第四、第五只猴子都是这样做的，问海滩上原来最少有多少个桃子？
2. 下面的程序是一个水果计价系统，程序中定义了一个结构体FRUIT来保存水果名称和单价，程序中含有多处错误，请修改程序中的错误，但不能改变程序结构。

#include <stdio.h>

#include <string.h>

typedef struct tagFRUIT

{

char szFuitName[20];//水果名称

double dbPrice;//单价

}FRUIT,\*PFRUIT;

FRUIT\* GetFruitArry(int kinds)

{

FRUIT fruit[] = {{"apple",8.0},{"pear",5.2},{"grape",7.8},{"banana",12.7},{"orange",4.3}};

kinds = sizeof(fruit)/sizeof(FRUIT);

return fruit;

}

void PrintFruitInformation()

{

int nFruitKinds = 0;

FRUIT\* pFruitArry = GetFruitArry(nFruitKinds);

int i;

printf("There are %d kinds of fruit here:\n",nFruitKinds);

for(i = 0;i<nFruitKinds;i++)

{

printf("%s price :%2.1f ($/pounds)\n",pFruitArry[i].szFuitName,pFruitArry[i].dbPrice);

}

}

int ChooseFruit(FRUIT fruit)

{

int bChoose = 0;

int nFruitKinds = 0;

FRUIT\* pFruitArry = GetFruitArry(nFruitKinds);

int i;

printf("Please input fruitName:\n");

scanf("%s",fruit.szFuitName);

FRUIT\* pPtrFruit = pFruitArry;

for (i =0 ;i<nFruitKinds;i++, pPtrFruit+= sizeof(FRUIT))

{

if(strcmp(pPtrFruit->szFuitName , fruit.szFuitName) == 0)

{

fruit = \*pPtrFruit;

bChoose = 1;

break;

}

}

if(!bChoose)

{

printf("can't find fruit '%s',please input again\n",fruit.szFuitName);

}

return bChoose;

}

int main()

{

FRUIT fruit;

PrintFruitInformation();

while( 0 == ChooseFruit(fruit));

printf("Please enter the purchase quantity (pounds):\n");

double pounds = 0;

do

{

fflush(stdin);

scanf("%f",&pounds);

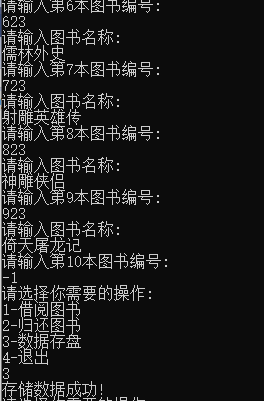
} while (pounds<=0.0000001);

double dbMoney =pounds\*fruit.dbPrice;

printf("You need pay %3.2f $ for %3.1f pounds of %s",dbMoney,pounds,fruit.szFuitName);

return 0;

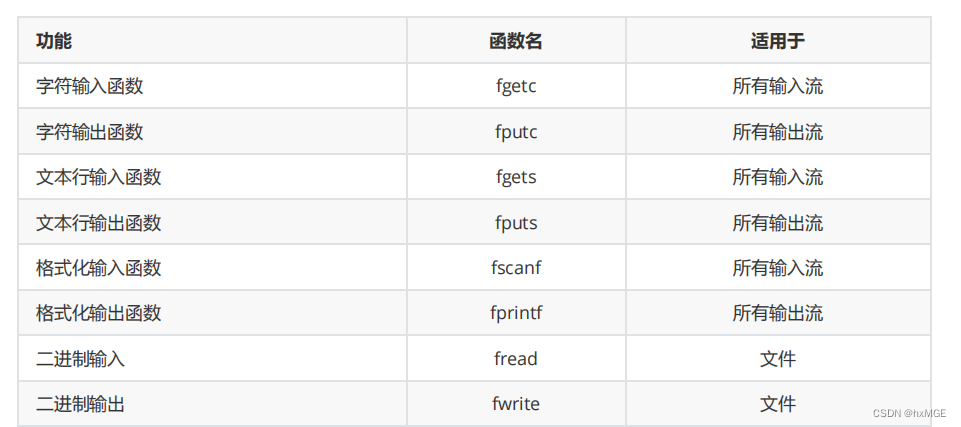
}

3. 小张同学为了提升班级同学的阅读兴趣，收集了不超过100本书，成立了一个小型图书馆，请设计一个程序，帮助小张对图书进行管理。

1. 每本书的信息包括书名（不超过30个字符），图书编号(大于0的整数)，借阅者学号（大于0的整数，如果图书未借出，则借阅者学号为0）。用结构体表示一本书的信息
2. 程序启动时，检查 d:\books.dat 文件，如果打开失败，表明目前尚未录入书籍信息，提示用户录入书籍信息，当录入书籍数量达到100或录入书籍的编号小于等于0且已经录入了正确数据时，结束录入；如果打开文件成功，读入存储图书的信息。
3. 显示如图所示操作菜单（借阅图书、归还图书、数据存盘、退出）
4. 根据输入菜单选择，进入相应的功能（借阅、归还、存储、退出）。
5. 借阅图书功能：要求输入借阅者学号（需要进行防错处理），每个人限借3本图书，如果查询当前借阅者借阅图书数目大于3时，提示已超出借书数量的信息，并结束借书。否则，提示是否凭编号借书，如果是凭编号借书，则根据编号查询图书信息，如果没有查到，输出“查无此书，借书失败”的提示信息，如果查到，则修改该图书的借阅者编号信息，提示借书成功（要求打印出书名）。如果是凭书名借书，则提示输入书名，并按照书名查询该书，借书步骤和按照编号借书步骤一样。
6. 归坏图书功能：提示输入归还图书的编号，查询输入的编号是否是本馆图书的编号，如果不是，输出非本馆图书的提示信息，如果是，则修改图书的借阅者编号信息为非借阅学号 0，并提示已归还图书（打印书名）。
7. 存储功能： 把图书信息存盘时存在d:\books.dat 文件

fopen





int fputc ( int ch, FILE \*fp );

ch是要输出的字符(输入的字符会转为ascii码值)，fp是流（文件，stdin键盘，stdout屏幕等）

如果没有发生错误，则返回被写入的字符。如果发生错误，则返回 EOF，并设置错误标识符。

int fgetc (FILE \*fp);

fp是流，从哪写入（文件，键盘，屏幕等）

该函数返回值以无符号 char 强制转换为 int 的形式返回读取的字符，如果到达文件末尾或发生读错误，则返回 EOF。

示例1：fputc

每次从ch中向文件里输出一个字符

int main()

{

FILE\* fp = fopen("text.txt","w");w是写操作，会在指定路径创建这个文件，即使有这个文件也会被重新创建覆盖

if(fp == NULL)

{

perror("fopen");

return 1;

}

char ch = 'a';

for(ch = 'a';ch<='z';ch++)

fputc(ch,fp);

每次从ch中向文件里输出一个字符，文件中的位置指针就指向下一个位置

fclose(fp);

fp = NULL;

}

示例2：fgetc

每次从文件读取一个字符存到ch中

int main()

{

FILE\* fp = fopen("text.txt","r");r是只读操作

if(fp == NULL)

{

perror("fopen");

return 1;

}

char ch = 0;

while( (ch = fgetc(fp)) != EOF)

printf("%c\n",ch);

每次通过函数从文件中读取的字符，并返回读取的字符存入ch，并通过字符型式打印

这里面有一个位置指针，每次读取一个字符，文件中的位置指针就会指向下一个字符，所以他是顺序读

直到文件末尾或出错，则返回EOF

fclose(fp);

fp = NULL;

}

fgets 、fputs

char \*fgets(char \*str, int n, FILE \*stream)

参数1：这是指向一个字符数组的指针，该数组存储了要读取的字符串

参数2：这是要读取的最大字符数（包括最后的空字符）。通常是使用以 str 传递的数组长度。

参数3：这是指向 FILE 对象的指针，该 FILE 对象标识了要从中读取字符的流。

返回值：如果成功，该函数返回相同的 str 参数。如果到达文件末尾或者没有读取到任何字符，str 的内容保持不变，并返回一个空指针

如果发生错误，返回一个空指针。

并且fgets会保存换行符

int fputs(const char \*str, FILE \*stream)

参数1： 这是一个数组，包含了要写入的以空字符终止的字符序列。

参数2： 这是指向 FILE 对象的指针，该 FILE 对象标识了要被写入字符串的流。

返回值：该函数返回一个非负值，如果发生错误则返回 EOF。

示例1：fputs

每次向流中存入一行字符串（包括\0）

int main()

{

FILE\* fp = fopen("text.txt","w");

if(fp == NULL)

{

perror("fopen");

return 1;

}

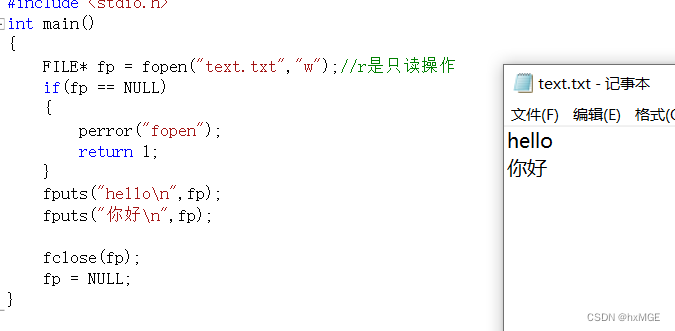
printf("%d",fputs("hello\n",fp));

printf("%d",fputs("你好\n",fp));

fclose(fp);

fp = NULL;

}



示例2：fgets

每次从流中读取一行字符串，最多读取 n个字符（包括\0）

int main()

{

char ch[256] = {0};

FILE\* fp = fopen("text.txt","r");

if(fp == NULL)

{

perror("fopen");

return 1;

}

第一行

fgets(ch,256,fp); 256的意思是一行中最大读取数是256个，实际读取到\0就结束了

printf("%s",ch); 所以第一行最大读取数实际就是256-1个，因为有\0

第二行

fgets(ch,256,fp);

fputs(ch,stdout); stdout的意思是屏幕，就是从ch中的字符串输出到屏幕上

或者

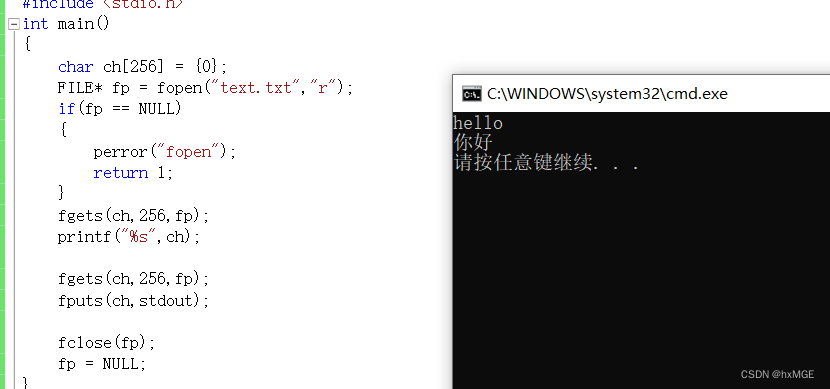
while(fgets(ch,256,fp) != NULL)

fputs(ch,stdout);

当文件位置指针到末尾时，也就是读取到了末尾，就会返回NULL

fclose(fp);

fp = NULL;

}

fprintf、fscanf

int fprintf(FILE \*stream, const char \*format, ...)

比起平常使用的printf多了一个FILE\* stream，意思是输出到哪个流上

int fscanf(FILE \*stream, const char \*format, ...)

比起平常使用的scanf多了一个FILE\* stream，意思是从哪个流输入数据

示例1：fprintf

struct stu

{

char name[20];

int age;

double b;

};

int main()

{

struct stu ps = {{"张三"},20,98.8};

FILE\* fp = fopen("text.txt","w");

if(fp == NULL)

{

perror("fopen");

return 1;

}

fprintf(fp,"%s %d %lf\n",ps.name,ps.age,ps.b);

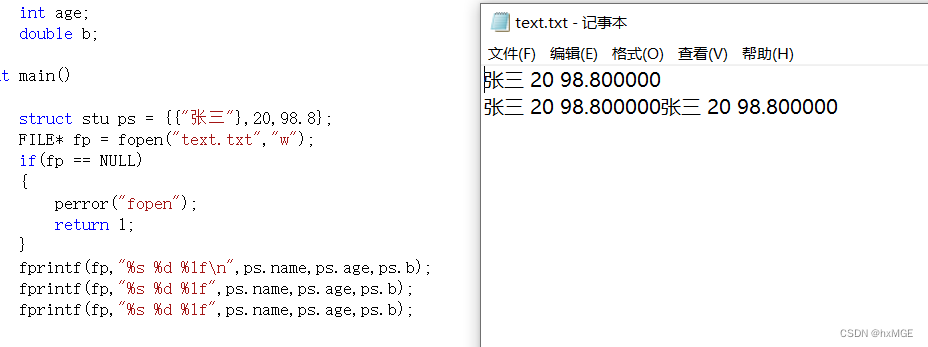
fprintf(fp,"%s %d %lf",ps.name,ps.age,ps.b);

fprintf(fp,"%s %d %lf",ps.name,ps.age,ps.b);

fclose(fp);

fp = NULL;

}



示例2：fscanf

struct stu

{

char name[20];

int age;

double b;

};

int main()

{

struct stu ps = {{"张三"},20,98.8};

FILE\* fp = fopen("text.txt","r");

if(fp == NULL)

{

perror("fopen");

return 1;

}

struct stu pa = {0};

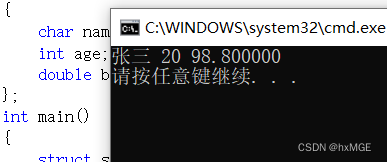
fscanf(fp,"%s %d %lf",pa.name,&(pa.age),&(pa.b));

printf("%s %d %lf\n",pa.name,pa.age,pa.b);

fclose(fp);

fp = NULL;

}



使用上述两个函数要注意：  
1.fprintf想在文件中换行，需要手动输入\n  
2.fscanf不要忘记除了字符串以外的取地址  
3.使用这两个函数一定要格式对应，怎么存的格式就怎么读出，且顺序一定要一样

seek

文件中会有一个位置指针，我们可以通过更改位置指针来达到随机读写

int fseek ( FILE \* stream, long int offset, int origin )

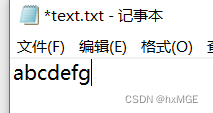
参数1：这是指向 FILE 对象的指针，该 FILE 对象标识了流。

参数2：这是相对 whence 的偏移量，以字节为单位。

参数3：SEEK\_CUR-文件指针当前位置、SEEK\_END-文件指针末尾的地址、SEEK\_SET—文件指针开始位置

返回值：如果成功，则该函数返回零，否则返回非零值。

例1：



已知已经创建了一个文件

如何只打印c，请看代码：

int main()

{

char ch = 0;

FILE\* fp = fopen("text.txt","r");

if(fp == NULL)

{

perror("fopen");

return 1;

}

fseek(fp,2,SEEK\_SET);

ch = fgetc(fp);

fputc(ch,stdout);

当要读到f时

fseek(fp,-2,SEEK\_END); //这里-2说明,edn是在g后面的位置

ch = fgetc(fp);

fputc(ch,stdout);

fclose(fp);

fp = NULL;

}

示例2：更改指定位置的内容

int main()

{

char ch = 0;

FILE\* fp = fopen("text.txt","w");

if(fp == NULL)

{

perror("fopen");

return 1;

}

fputc('a',fp);

fputc('b',fp);

fputc('c',fp);

fseek(fp,-2,SEEK\_CUR);

先往文件中写入 a，b，c

在改变位置指针到b的位置

将b替换为w

fputc('w',fp);

fclose(fp);

fp = NULL;

}

feof(fp) 就是判断fp是否已经读取了EOF字符。如果已读取，返回true值，所以在调用feof(fp) 之前，都应该先调用读文件的函数： fgets(ar, 50,fid)、fgetc(fid)，然后再判断就OK。