API文档

GenerateData（测试用）

创建书本信息

调用`GenerateData::CreateBook(vector<Book> book);`

传入**书本结构体**组成的vector，结构体定义在`Data.h`中

其中有三次写入，一次是对书本信息的写入，输出给`Book.dat`文件，一次是生成编号索引，输出给`BookIdIndex.dat`，最后是生成书名索引，输出为`BookNameIndex.dat`

SearchTool

按照书名搜索

`vector<pair<Book, long>> SearchTool::SearchBookName(string name)`

只需要传入需要搜索的书名即可。此函数为模糊搜索，将传入的书名根据长度切分，最多三等分，然后调用多线程进行搜索。

具体搜索行为在`void SearchTool::SubSearch(std::string name)`中。如果 name 是书名中的字符串，如 name = "C", 书名：“C++程序设计”。则将对应书本的地址存入全局变量`vector<long> index`中。

搜索结束后返回`SearchBookName(string name)`，将`index`中的重复地址去除，然后根据得到的地址在书本文件中直接获取书本信息。将得到的信息存入`vector<Book> book`中返回

结束后返回 book组成的vector

按照编号搜索

`pair<Book, long> SearchTool::SearchBookId(char id[avglen])`

传入一个书本编号进行搜索，主要用于管理员界面。

同样返回一个包含书本信息的`vector<Book>`，不同的是，返回值只包含最多一本书。因为本项目默认书本编号不重复，此函数在查到符合的编号后便返回。

按照ISBN搜索

`pair<Book, long> SearchBookISBN(char id[avglen])`

传入对应的ISBN编号就可以搜索，如果没有，返回的地址将是 -1.

查找某类书所有书本

需要传入这类书在`Book.dat`中的地址，然后查找

返回一个`vector<pair<BookIdIndex, long>>`

工具

Book SearchTool::bookup(char * content)

此函数将传入的字符串解释成一本书的信息，然后返回一个`struct Book`。

void SearchTool::split(char * s1, char *s2, int offset, int count)

将 s1 从 offset 开始，一共 count 个字符传入 s2。

bool SearchTool::find(char s1[], char s2[])

在 s1 中寻找是否有与 s2 匹配的子串。需要注意的是，只搜索 s1 从 0 到 avglen - 1的下标，avglen是书本字符串类型信息的长度，因为此函数通常用于搜索中文名，所以设置固定长度避免中途遇到`\0`而意外结束搜索。

void bookdown(Book* book, fstream*io)

传入一本书的信息与输入输出流指针，利用 io 将 book 输入书本文件中

void ltob(long n, char * s)

将长整型转换为字符串，字符串长度为`sizeof(long)`

void itob(int n, char * s)

将整型转换为字符串，字符串长度为`sizeof(int)`

void ftob(float n, char *s)

将浮点型转换为字符串，字符串长度为`sizeof(float)`

long btol(char *s)

将字符串解释为长整型

int btoi(char *s)

解释为整型

float btof(char *s)

解释为浮点型

BookIdIndex idup(char *content)

将字符串内的内容转换为`BookIdIndex`类型