技术文档-搜索

10月1日

确定书本存储的数据结构

```
struct Book
{
    string id;
    string name;
    string author;
    string press;       //出版社
    string category;    //学科
    long pubdate;       //出版日期
    bool type;          //借阅状态
    int num;            //借阅次数
    float price;
};
// 书名索引
struct BookNameIndex
{
    char name[avglen];
    long index;
};
```

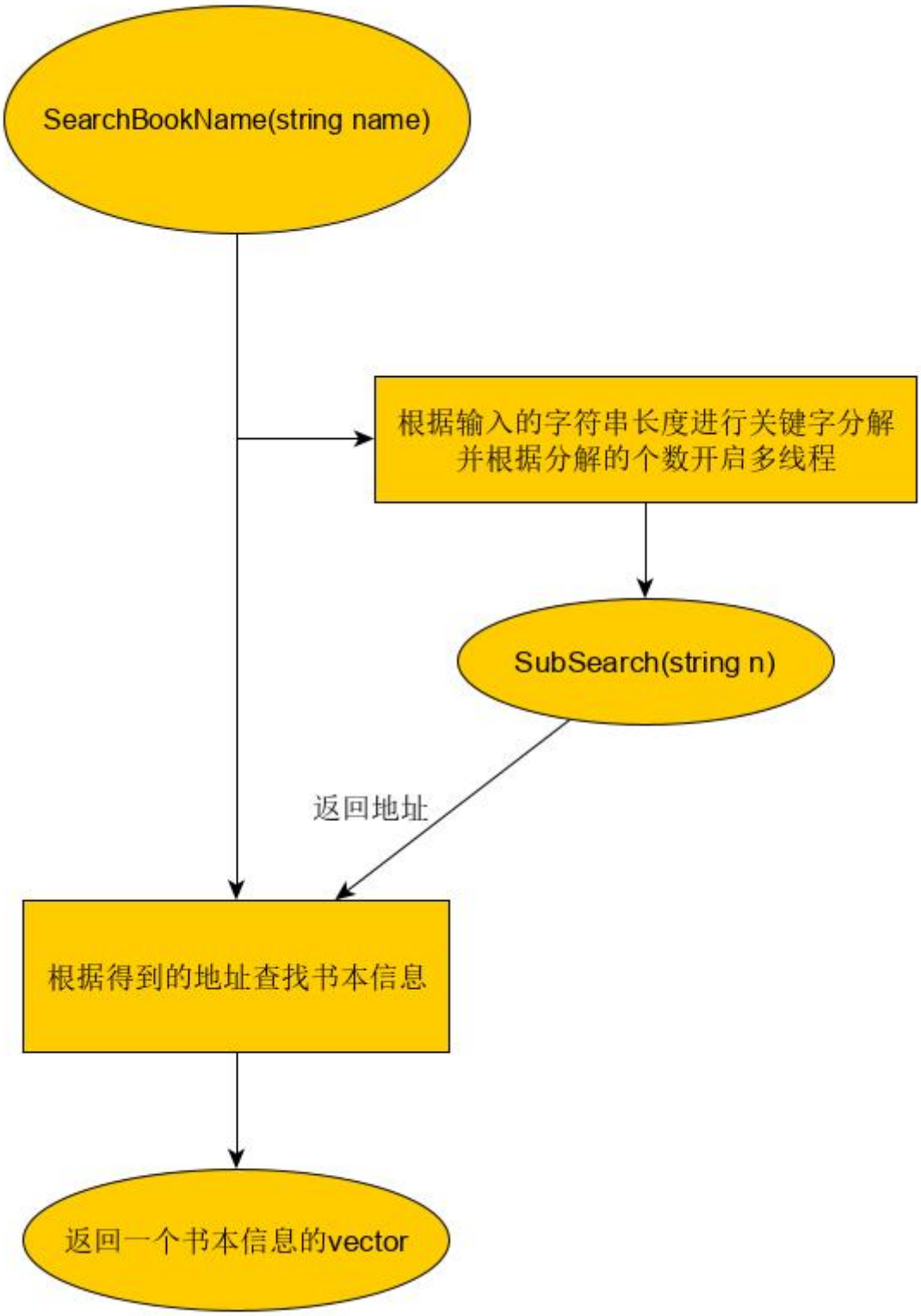
实现利用书名索引进行搜索

```
class SearchTool
{
private:
    static string bookName;                  //书本信息文件名
    static void SubSearch(std::string n);    //实现在书名索引内搜索关键字
    static bool find(char s1[], char s2[]); //判断s1是否包含s2

public:
    // 静态操作方法
    static std::vector<Book> SearchName(std::string name);  //按名字查找，已实现
    static std::vector<Book> SearchNum(int num);            //按编号查找，待实现
    static std::vector<Book> SearchAuthor(char author[10]); //按作者查找，待实现

    //分别是长整型、整型、浮点型与字节数组的转换
    static void ltob(long n, char * s);
    static void itob(int n, char * s);
    static void ftob(float n, char *s);
    static long btol(char *s);
    static int btoi(char *s);
    static float btof(char *s);


public:
    SearchTool();
```

```
    ~SearchTool();
};
```

目前已经实现按书名进行搜索，但并未进行大规模数据测试

算法示意图



接下来实现按照编号进行搜索

10月4日

完成按照编号索引，数据格式如下：

```
struct BookNumIndex
{
    int num;
    long index;
};
```

编号查找算法较为简单

```
std::vector<Book> SearchTool::SearchBookNum(int num)
{
    ifstream io("BookNumIndex.dat", ios::in | ios::binary);
    std::vector<Book> book;
    int size = sizeof(int) + sizeof(long);
    char tmp[500];
    char n[20];
    itob(num, n);//将编号转换为字符串

    while (!io.eof())
    {
        io.read(tmp, size);
        tmp[size] = '\0';
        if (cmp(tmp, n, sizeof(int)))//找到匹配的之后直接根据地址获取书本信息
        {
            split(tmp, n, sizeof(int), sizeof(long));
            long idx = btol(n);
            ifstream in(bookName, ios::in | ios::binary);
            in.seekg(idx, ios::beg);
            in.read(tmp, bookInfoSize);
            book.push_back(bookup(tmp));
            return book;
        }
    }
    return book;
}
```

直接在编号索引表中查找匹配传入的编号，找到之后立即根据地址在书本文件中查找对应的书本信息

另外更新比较函数用于判断 s1 、s2 在 size 的长度内是否相同

```
bool cmp(char *s1, char *s2, int size);
```

10月12日

按照书名搜索速度测试完毕（100万本，书名索引文件）

项目部分完成功能。

1. 按照编号索引
2. 完成删除书本功能 `DeleteTool` （待测试）
3. 完成`void bookdown(Book*, fstream *)`函数用于写入书本信息

测试部分完成功能

完成`GenerateData`类，用于生成测试数据

注意：将主要公用数据迁移至`Data.h`中方便开发，将工具函数迁移至`Tool.cpp`中

另附 API 文档在 `API文档.md` 中

10月19日

微调数据结构

```
struct BookIdIndex
{
    char id[avglen];             //书本编号
    long index;     //地址
    char isBorrowed[1]; //是否借出
};
```

编号索引现在多了一个是否借阅的指标

Book 取消 type 这个数据成员

完善修改工具类

现在操作方式是：修改、删除、增加某类书；修改、删除、增加某本书

查询操作

1. 现在查询操作将传出书本、地址：`vector<pair<Book, long>>`
2. 增加了针对ISBN的查询
3. 增加了对某类书的所有编号的查询