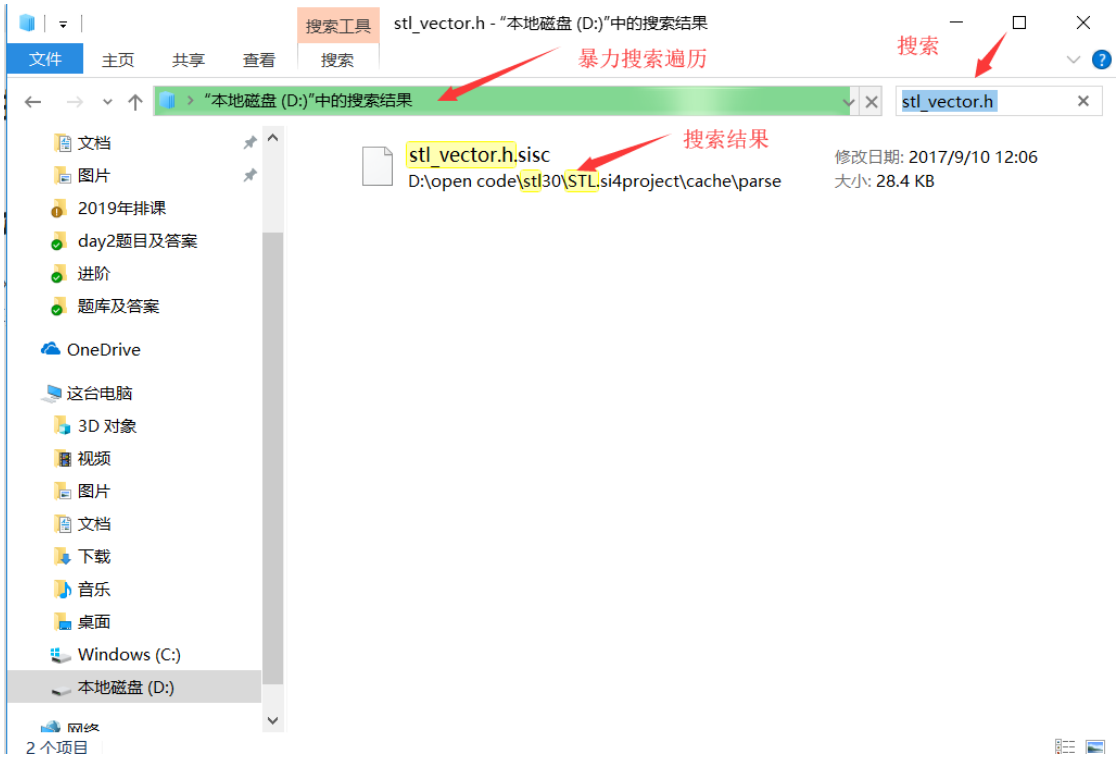
快速搜索文档神器

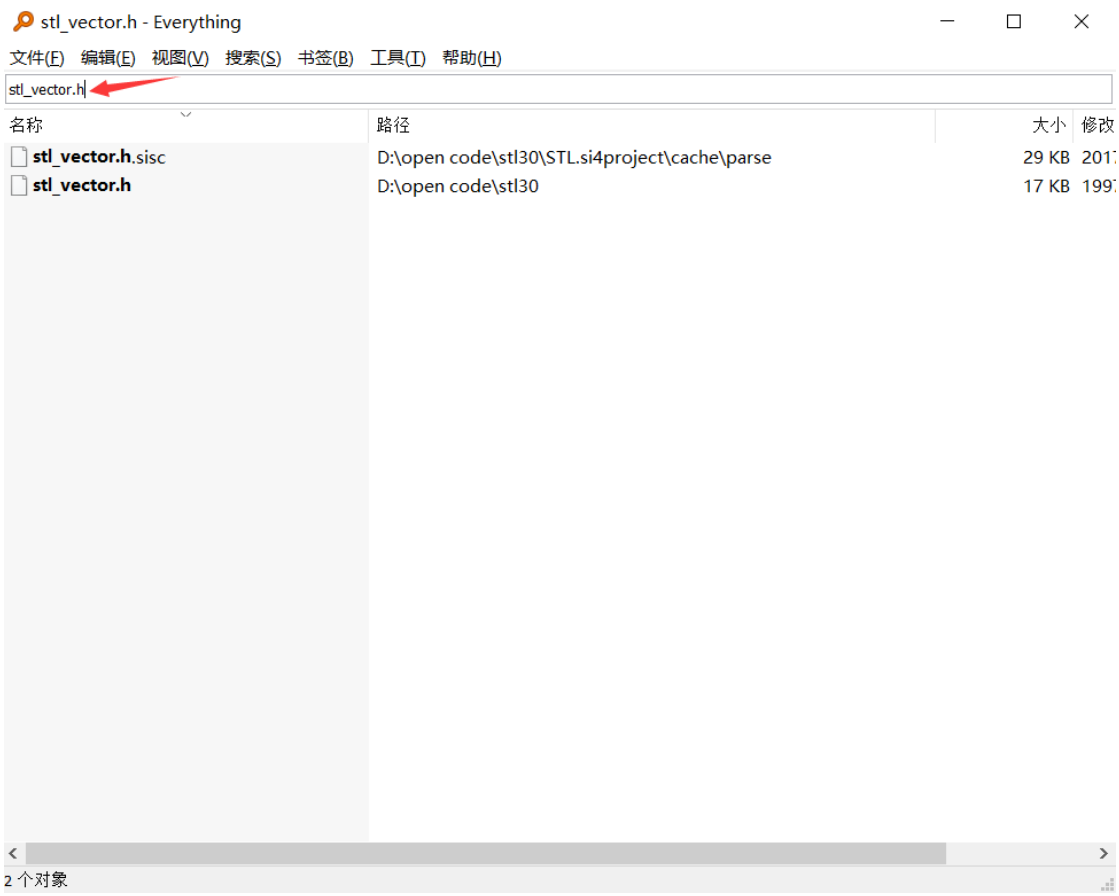
一、项目分析

1. 项目调研实现背景

> linux环境下有非常好用的find命令，查找文档非常的便捷高效，而windows下文件夹框下的默认搜索是搜索时再进行暴力遍历查找，非常的慢。



> windows下有一个神器软件解决了这个问题，叫everything，是将文档信息检索以后，提前存储到数据库，查找时在数据库进行搜索，速度快了很多。[everything软件下载]



2.项目需求

1.支持文档普通搜索
2.支持拼音全拼搜索
3.支持拼音首字母搜索
4.支持搜索关键字高亮

3.开发环境

1.编译器 : VS2013 / 控制应用平台
2.编程语言 : C++ / C++11
2.数据库 : sqlite3

4.项目涉及的知识点

1.数据库操作： （sqlite安装，创建数据库，创建表，插入数据，删除数据，创建索引，查询数据(条件查询、 模糊查询))

2.静态库和动态库：静态库和动态的制作，动态库和动态的使用

3.设计模式（单例模式）

4.多线程

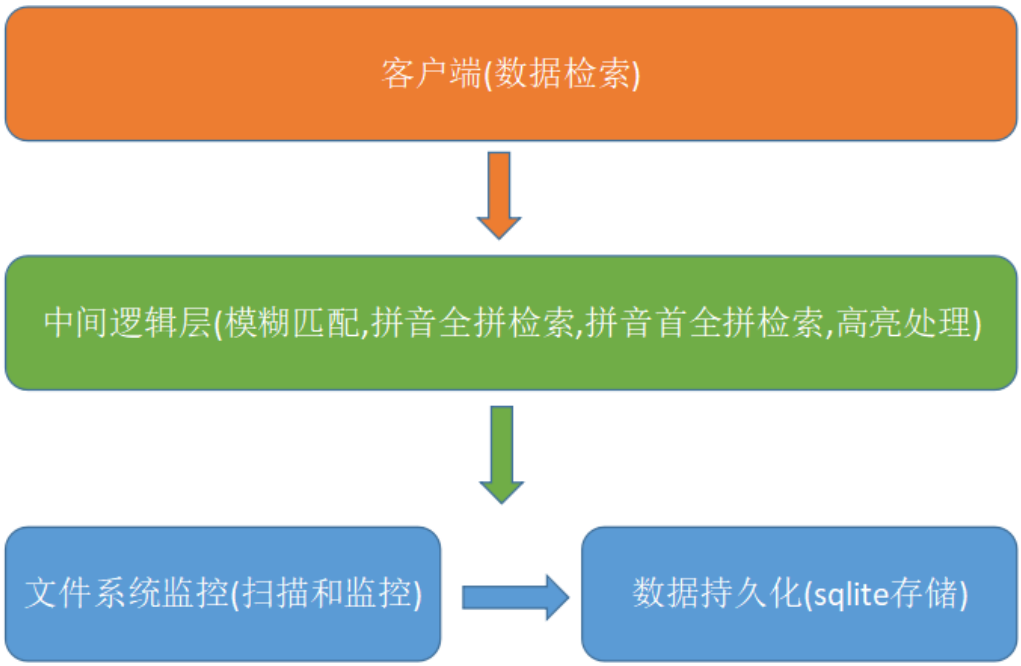
5.同步机制(互斥量)

6.日志

7.汉字与拼音的转换

二、项目设计

1.项目实现目标及框架

> **1.1实现目标：**
>
> 咋们本节课的目标就是造个轮子，写一个类似everything的搜索文档神器出来玩一玩。另外可以支持一些它没有的小功能，比如拼音搜索。^^
>
> 当然我们还可以扩展实现一些everything不支持的功能，比如：拼音搜索等等
>
> **1.2框架设计：**
>
> 
>
>
> **1.3代码框架：**

```
公共模块     ：  Common.h
系统工具模块 ：  Sysutil.h Sysutil.cpp
数据管理模块 ：  DataManager.h DataManager.cpp
扫描管理模块 ：  ScanManager.h ScanManager.cpp
系统驱动模块 ：  DocFastSearchTool.cpp
```

2.文件系统监控

> everything在实现这个模块时，使用了扫描+监控的实现方式，这两种方式是一种互补的方式。
>
> 1. 文件系统监控是利用系统文件系统的接口可以监控某个目录下的文档变化，有点是效率高实时性强，缺点是监控是实时的，如果在监控程序没有启动期间的，文档的变化无法获取。
> 2. 文件系统扫描是通过系统接口，遍历获取目录下的所有文档跟数据库中的文档进行对比，获取文档变化，优点是监控程序启动前，变化的文档也能对比出来，缺点是性能低实时性不强。
>
> 我们这里呢，为了简单一点，我们使用了简单粗暴的扫描。如有需要大家可以下去扩展实现可以加上监控。

3.数据持久化

> 数据持久化我们使用了轻量级的一个数据库sqlite管理，使用sqlite需要去下载sqlite的源码或者源码。

我们的数据库的表很简单，只需要一张表

```
create table if not exists tb_doc (id INTEGER PRIMARY KEY autoincrement, doc_path text, doc_name text, doc_name_pinyin text, doc_name_initials text);
```

4.中间逻辑层

everything没有实现拼音相关的搜索功能，实际中很多应用软件的搜索部分都实现了这个功能，这里我们类比我们常用的软件qq实现了拼音全拼和拼音首字母并且高亮的功能。

4.1模糊匹配

使用数据库的like实现模糊匹配检索。


```
请输入搜索关键字:比特科技
文件名                                                  路径
0-比特科技-排课计划                                     D:\bite\1-教学服务团队
1-比特课件                                              D:\bite\1-教学服务团队
比特科技排课原则指导.md                                 D:\bite\1-教学服务团队\0-比特科技-排课计划
比特科技试听课-第一讲.md                                D:\bite\1-教学服务团队\1-比特课件\0.试听课
比特科技试听课-第一讲.pdf                               D:\bite\1-教学服务团队\1-比特课件\0.试听课
比特科技编程新手训练营：授课体系.pdf                    D:\bite\1-教学服务团队\1-比特课件\1.C语言课件\编程新手训练
西安比特科技Java方向课程体系-v1.xmind                   D:\bite\1-教学服务团队\1-比特课件\8.Java课件\JavaSE课件\辅助材料
比特科技杯编程大赛（初级组）题目.md                     D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（初级组）题目.pdf                    D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（高级组）题目.md                     D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（高级组）题目.pdf                    D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技编程大赛初级组(复赛).md                         D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛初级组(复赛).pdf                        D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛高级组(复赛).md                         D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛高级组(复赛).pdf                        D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技校招论坛                                        D:\bite\1-教学服务团队\12-校园招聘跟进
58同城 - 比特科技.pdf                                   D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
58同城 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
CVTE面试题 - 比特科技提供.pdf                           D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
Momenta-比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
阿里巴巴 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
阿里巴巴-比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
爱数 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
百度 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
百度 -比特科技.pdf                                      D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
触宝 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
滴滴 -比特科技制作.pdf                                  D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
地平线机器人 - 比特科技提供.pdf                         D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
多益网络 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
瓜子-比特科技提供.pdf                                   D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
广联达 -比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
海康威视 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
好未来 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
欢聚时代 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
吉比特 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
吉比特 -比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
金山WPS - 比特科技提供.pdf                              D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
金山WPS -比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
今日头条 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
```

4.2拼音全拼搜索

存储时将文件名转换成一个拼音全拼存在数据库表的doc_name_pinyin字段中，搜索时也将关键字转换成拼音，然后使用数据库的模糊匹配搜索


```
请输入搜索关键字:bitekeji
文件名                                                  路径
0-比特科技-排课计划                                     D:\bite\1-教学服务团队
1-比特课件                                              D:\bite\1-教学服务团队
比特科技排课原则指导.md                                 D:\bite\1-教学服务团队\0-比特科技-排课计划
比特科技试听课-第一讲.md                                D:\bite\1-教学服务团队\1-比特课件\0.试听课
比特科技试听课-第一讲.pdf                               D:\bite\1-教学服务团队\1-比特课件\0.试听课
比特科技编程新手训练营：授课体系.pdf                    D:\bite\1-教学服务团队\1-比特课件\1.C语言课件\编程新手训练
西安比特科技Java方向课程体系-v1.xmind                   D:\bite\1-教学服务团队\1-比特课件\8.Java课件\JavaSE课件\辅助材料
比特科技杯编程大赛（初级组）题目.md                     D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（初级组）题目.pdf                    D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（高级组）题目.md                     D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（高级组）题目.pdf                    D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技编程大赛初级组(复赛).md                         D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛初级组(复赛).pdf                        D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛高级组(复赛).md                         D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛高级组(复赛).pdf                        D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技校招论坛                                        D:\bite\1-教学服务团队\12-校园招聘跟进
58同城 - 比特科技.pdf                                   D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
58同城 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
CVTE面试题 - 比特科技提供.pdf                           D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
Momenta-比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
阿里巴巴 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
阿里巴巴-比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
爱数 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
百度 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
百度 -比特科技.pdf                                      D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
触宝 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
滴滴 -比特科技制作.pdf                                  D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
地平线机器人 - 比特科技提供.pdf                         D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
多益网络 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
瓜子-比特科技提供.pdf                                   D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
广联达 -比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
海康威视 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
好未来 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
欢聚时代 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
吉比特 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
吉比特 -比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
```

4.3拼音首字母搜索

存储时将文件名转换成一个拼音首字母存在数据库表的doc_name_initials字段中，搜索时也将关键字转换成拼音首字母，然后使用数据库的模糊匹配搜索


```
请输入搜索关键字:btkj
文件名                                                  路径
0-比特科技-排课计划                                     D:\bite\1-教学服务团队
1-比特课件                                              D:\bite\1-教学服务团队
比特科技排课原则指导.md                                 D:\bite\1-教学服务团队\0-比特科技-排课计划
比特科技试听课-第一讲.md                                D:\bite\1-教学服务团队\1-比特课件\0.试听课
比特科技试听课-第一讲.pdf                               D:\bite\1-教学服务团队\1-比特课件\0.试听课
比特科技编程新手训练营：授课体系.pdf                    D:\bite\1-教学服务团队\1-比特课件\1.C语言课件\编程新手训练
西安比特科技Java方向课程体系-v1.xmind                   D:\bite\1-教学服务团队\1-比特课件\8.Java课件\JavaSE课件\辅助材料
比特科技杯编程大赛（初级组）题目.md                     D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（初级组）题目.pdf                    D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（高级组）题目.md                     D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技杯编程大赛（高级组）题目.pdf                    D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\初赛
比特科技编程大赛初级组(复赛).md                         D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛初级组(复赛).pdf                        D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛高级组(复赛).md                         D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技编程大赛高级组(复赛).pdf                        D:\bite\1-教学服务团队\10-校企合作\编程大赛\2018年\复赛
比特科技校招论坛                                        D:\bite\1-教学服务团队\12-校园招聘跟进
58同城 - 比特科技.pdf                                   D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
58同城 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
CVTE面试题 - 比特科技提供.pdf                           D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
Momenta-比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
阿里巴巴 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
阿里巴巴-比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
爱数 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
百度 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
百度 -比特科技.pdf                                      D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
触宝 - 比特科技提供.pdf                                 D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
滴滴 -比特科技制作.pdf                                  D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
地平线机器人 - 比特科技提供.pdf                         D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
多益网络 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
瓜子-比特科技提供.pdf                                   D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
广联达 -比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
海康威视 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
好未来 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
欢聚时代 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
吉比特 - 比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
吉比特 -比特科技提供.pdf                                D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
金山WPS - 比特科技提供.pdf                              D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
金山WPS -比特科技提供.pdf                               D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
今日头条 - 比特科技提供.pdf                             D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
今日头条 -比特科技提供.pdf                              D:\bite\1-教学服务团队\15-笔试面试真题\面试真题
```

4.4高亮处理

高亮处理需要对搜索出的关键字高亮标记处理，这里如果是直接的模糊匹配，比较简单，就是一个子串匹配，但是拼音全拼搜索和首字母搜索需要使用一套逻辑算法来处理。

三、项目实现

1.系统工具模块实现

1.1 目录扫描

```
主要涉及类型与函数:
struct _finddata_t
{
unsigned attrib;
time_t time_create;
time_t time_access;
time_t time_write;
_fsize_t size;
char name[_MAX_FNAME];
};
long _findfirst( char *filespec, struct _finddata_t *fileinfo );
int _findnext( long handle, struct _finddata_t *fileinfo );
int _findclose( long handle );
```

```
static void DirectoryList(const std::string& path,
std::vector<std::string>& subfiles, std::vector<std::string>& subdirs)
{
    _finddata_t file;
    std::string path_ = path + "\\*.*";
    intptr_t handle = _findfirst(path_.c_str(), &file);
    if (handle == -1)
    {
        ERROE_LOG("_findfirst:%s", path.c_str());
    }
    do {
        //判断是否有子目录
        if (file.attrib & _A_SUBDIR)
        {
            //判断是否为"."当前目录，".."上一层目录
            if ((strcmp(file.name, ".") != 0) && (strcmp(file.name,
"..") != 0))
            {
                subdirs.push_back(file.name);
            }
        }
        else
        {
            subfiles.push_back(file.name);
        }
    } while (_findnext(handle, &file) == 0);

    _findclose(handle);
}
```

1.2 日志模块

```
日志系统 1:
#ifndef __TRACE__
//  #define __TRACE__
#endif
```

```
5
6  #ifndef __DEBUG__
7      #define __DEBUG__
8  #endif
9
10 ////////////////////////////////////////////////////////////////////
   ///
11 static std::string GetFileName(const std::string& path)
12 {
13     char ch = '/';
14
15 #ifdef _WIN32
16     ch = '\\';
17 #endif
18     size_t pos = path.rfind(ch);
19     if (pos == std::string::npos)
20         return path;
21     else
22         return path.substr(pos + 1);
23 }
24
25 //用于调试追溯的trace log
26 inline static void __TraceDebug(const char* filename, int line,
   const char* function, const char* format, ...)
27 {
28 #ifdef __TRACE__
29     //输出调用函数的信息
30     fprintf(stdout,"[TRACE]
   [%s:%d:%s]:",GetFileName(filename).c_str(), line, function);
31
32     //输出用户打的trace信息
33     va_list args;
34     va_start(args,format);
35     vfprintf(stdout,format, args);
36     va_end(args);
37
38     fprintf(stdout,"\n");
39 #endif
40 }
41
42 inline static void __ErrorDebug(const char* filename, int line,
   const char* function, const char* format, ...)
43 {
44 #ifdef __DEBUG__
45     //输出调用函数的信息
46     fprintf(stdout,"[ERROR]
   [%s:%d:%s]:",GetFileName(filename).c_str(), line, function);
47
48     //输出用户打的trace信息
49     va_list args;
50     va_start(args,format);
51     vfprintf(stdout,format, args);
52     va_end(args);
53
54     fprintf(stdout," errmsg:%s, errno:%d\n", strerror(errno),
   errno);
55 #endif
56 }
```

```
57
58 #define TRACE_LOG(...) \
59     __TraceDebug(__FILE__, __LINE__, __FUNCTION__, __VA_ARGS__);
60
61 #define ERROR_LOG(...) \
62     __ErrorDebug(__FILE__, __LINE__, __FUNCTION__, __VA_ARGS__);
63
```

1.3 汉字转拼音全拼

```
1  // 汉字转拼音全拼
2  /* CSDN：http://blog.csdn.net/csnd_ayo *
3  static std::string ChineseConvertPinYinAllSpell(const std::string&
   dest_chinese)
4  {
5  static const int spell_value[] = { -20319, -20317, -20304, -20295,
   -20292, -20283, -20265, -20257, -20242, -20230, -20051, -20036,
   -20032, -20026,
6      -20002, -19990, -19986, -19982, -19976, -19805, -19784, -19775,
   -19774, -19763, -19756, -19751, -19746, -19741, -19739, -19728,
7      -19725, -19715, -19540, -19531, -19525, -19515, -19500, -19484,
   -19479, -19467, -19289, -19288, -19281, -19275, -19270, -19263,
8      -19261, -19249, -19243, -19242, -19238, -19235, -19227, -19224,
   -19218, -19212, -19038, -19023, -19018, -19006, -19003, -18996,
9      -18977, -18961, -18952, -18783, -18774, -18773, -18763, -18756,
   -18741, -18735, -18731, -18722, -18710, -18697, -18696, -18526,
10     -18518, -18501, -18490, -18478, -18463, -18448, -18447, -18446,
   -18239, -18237, -18231, -18220, -18211, -18201, -18184, -18183,
11     -18181, -18012, -17997, -17988, -17970, -17964, -17961, -17950,
   -17947, -17931, -17928, -17922, -17759, -17752, -17733, -17730,
12     -17721, -17703, -17701, -17697, -17692, -17683, -17676, -17496,
   -17487, -17482, -17468, -17454, -17433, -17427, -17417, -17202,
13     -17185, -16983, -16970, -16942, -16915, -16733, -16708, -16706,
   -16689, -16664, -16657, -16647, -16474, -16470, -16465, -16459,
14     -16452, -16448, -16433, -16429, -16427, -16423, -16419, -16412,
   -16407, -16403, -16401, -16393, -16220, -16216, -16212, -16205,
15     -16202, -16187, -16180, -16171, -16169, -16158, -16155, -15959,
   -15958, -15944, -15933, -15920, -15915, -15903, -15889, -15878,
16     -15707, -15701, -15681, -15667, -15661, -15659, -15652, -15640,
   -15631, -15625, -15454, -15448, -15436, -15435, -15419, -15416,
17     -15408, -15394, -15385, -15377, -15375, -15369, -15363, -15362,
   -15183, -15180, -15165, -15158, -15153, -15150, -15149, -15144,
18     -15143, -15141, -15140, -15139, -15128, -15121, -15119, -15117,
   -15110, -15109, -14941, -14937, -14933, -14930, -14929, -14928,
19     -14926, -14922, -14921, -14914, -14908, -14902, -14894, -14889,
   -14882, -14873, -14871, -14857, -14678, -14674, -14670, -14668,
20     -14663, -14654, -14645, -14630, -14594, -14429, -14407, -14399,
   -14384, -14379, -14368, -14355, -14353, -14345, -14170, -14159,
21     -14151, -14149, -14145, -14140, -14137, -14135, -14125, -14123,
   -14122, -14112, -14109, -14099, -14097, -14094, -14092, -14090,
22     -14087, -14083, -13917, -13914, -13910, -13907, -13906, -13905,
   -13896, -13894, -13878, -13870, -13859, -13847, -13831, -13658,
23     -13611, -13601, -13406, -13404, -13400, -13398, -13395, -13391,
   -13387, -13383, -13367, -13359, -13356, -13343, -13340, -13329,
24     -13326, -13318, -13147, -13138, -13120, -13107, -13096, -13095,
   -13091, -13076, -13068, -13063, -13060, -12888, -12875, -12871,
```

```
    -12860, -12858, -12852, -12849, -12838, -12831, -12829, -12812, -12802, -12607, -12597, -12594, -12585, -12556, -12359, -12346,
    -12320, -12300, -12120, -12099, -12089, -12074, -12067, -12058, -12039, -11867, -11861, -11847, -11831, -11798, -11781, -11604,
    -11589, -11536, -11358, -11340, -11339, -11324, -11303, -11097, -11077, -11067, -11055, -11052, -11045, -11041, -11038, -11024,
    -11020, -11019, -11018, -11014, -10838, -10832, -10815, -10800, -10790, -10780, -10764, -10587, -10544, -10533, -10519, -10331,
    -10329, -10328, -10322, -10315, -10309, -10307, -10296, -10281, -10274, -10270, -10262, -10260, -10256, -10254 };
// 395个字符串，每个字符串长度不超过6
static const char spell_dict[396][7] = { "a", "ai", "an", "ang", "ao", "ba", "bai", "ban", "bang", "bao", "bei", "ben", "beng", "bi", "bian", "biao",
"bie", "bin", "bing", "bo", "bu", "ca", "cai", "can", "cang", "cao", "ce", "ceng", "cha", "chai", "chan", "chang", "chao", "che", "chen",
"cheng", "chi", "chong", "chou", "chu", "chuai", "chuan", "chuang", "chui", "chun", "chuo", "ci", "cong", "cou", "cu", "cuan", "cui",
"cun", "cuo", "da", "dai", "dan", "dang", "dao", "de", "deng", "di", "dian", "diao", "die", "ding", "diu", "dong", "dou", "du", "duan",
"dui", "dun", "duo", "e", "en", "er", "fa", "fan", "fang", "fei", "fen", "feng", "fo", "fou", "fu", "ga", "gai", "gan", "gang", "gao",
"ge", "gei", "gen", "geng", "gong", "gou", "gu", "gua", "guai", "guan", "guang", "gui", "gun", "guo", "ha", "hai", "han", "hang",
"hao", "he", "hei", "hen", "heng", "hong", "hou", "hu", "hua", "huai", "huan", "huang", "hui", "hun", "huo", "ji", "jia", "jian",
"jiang", "jiao", "jie", "jin", "jing", "jiong", "jiu", "ju", "juan", "jue", "jun", "ka", "kai", "kan", "kang", "kao", "ke", "ken",
"keng", "kong", "kou", "ku", "kua", "kuai", "kuan", "kuang", "kui", "kun", "kuo", "la", "lai", "lan", "lang", "lao", "le", "lei",
"leng", "li", "lia", "lian", "liang", "liao", "lie", "lin", "ling", "liu", "long", "lou", "lu", "lv", "luan", "lue", "lun", "luo",
"ma", "mai", "man", "mang", "mao", "me", "mei", "men", "meng", "mi", "mian", "miao", "mie", "min", "ming", "miu", "mo", "mou", "mu",
"na", "nai", "nan", "nang", "nao", "ne", "nei", "nen", "neng", "ni", "nian", "niang", "niao", "nie", "nin", "ning", "niu", "nong",
"nu", "nv", "nuan", "nue", "nuo", "o", "ou", "pa", "pai", "pan", "pang", "pao", "pei", "pen", "peng", "pi", "pian", "piao", "pie",
"pin", "ping", "po", "pu", "qi", "qia", "qian", "qiang", "qiao", "qie", "qin", "qing", "qiong", "qiu", "qu", "quan", "que", "qun",
"ran", "rang", "rao", "re", "ren", "reng", "ri", "rong", "rou", "ru", "ruan", "rui", "run", "ruo", "sa", "sai", "san", "sang",
"sao", "se", "sen", "seng", "sha", "shai", "shan", "shang", "shao", "she", "shen", "sheng", "shi", "shou", "shu", "shua",
"shuai", "shuan", "shuang", "shui", "shun", "shuo", "si", "song", "sou", "su", "suan", "sui", "sun", "suo", "ta", "tai",
"tan", "tang", "tao", "te", "teng", "ti", "tian", "tiao", "tie", "ting", "tong", "tou", "tu", "tuan", "tui", "tun", "tuo",
"wa", "wai", "wan", "wang", "wei", "wen", "weng", "wo", "wu", "xi", "xia", "xian", "xiang", "xiao", "xie", "xin", "xing",
"xiong", "xiu", "xu", "xuan", "xue", "xun", "ya", "yan", "yang", "yao", "ye", "yi", "yin", "ying", "yo", "yong", "you",
"yu", "yuan", "yue", "yun", "za", "zai", "zan", "zang", "zao", "ze", "zei", "zen", "zeng", "zha", "zhai", "zhan", "zhang",
"zhao", "zhe", "zhen", "zheng", "zhi", "zhong", "zhou", "zhu", "zhua", "zhuai", "zhuan", "zhuang", "zhui", "zhun", "zhuo",
"zi", "zong", "zou", "zu", "zuan", "zui", "zun", "zuo" };
std::string pinyin;
```

```
// 循环处理字节数组
const int length = dest_chinese.length();
for (int j = 0, chrasc = 0; j < length;) {
// 非汉字处理
if (dest_chinese.at(j) >= 0 && dest_chinese.at(j) < 128) {
    pinyin += dest_chinese[j];
    // 偏移下标
    j++;
    continue;
}
// 汉字处理
chrasc = dest_chinese[j] * 256 + dest_chinese[j + 1] + 256;
if (chrasc > 0 && chrasc < 160) {
    // 非汉字
    pinyin += dest_chinese.at(j);
    // 偏移下标
    j++;
}
else {
    // 汉字
    for (int i = (sizeof(spell_value) / sizeof(spell_value[0]) - 1);
i >= 0; --i) {
        // 查找字典
        if (spell_value[i] <= chrasc) {
            pinyin += spell_dict[i];
            break;
        }
    }
    // 偏移下标 （汉字双字节）
    j += 2;
}
} // for end
return pinyin;
}
```

1.4汉字转拼音首字母

```
// 汉字转拼音首字母
static std::string ChineseConvertPinYinInitials(const std::string&
name)
{
    // 仅生成拼音首字母内容
    static int secPosValue[] = {
        1601, 1637, 1833, 2078, 2274, 2302, 2433, 2594, 2787, 3106,
3212,
        3472, 3635, 3722, 3730, 3858, 4027, 4086, 4390, 4558, 4684,
4925, 5249
    };

    static const char* initials[] = {
        "a", "b", "c", "d", "e", "f", "g", "h", "j", "k", "l", "m",
"n", "o",
        "p", "q", "r", "s", "t", "w", "x", "y", "z"
    };

    static const char* secondSecTable =
```

```
"CJWGNSPGCGNE[Y[BTYYZDXYKYGT[JNNJQMBSGZSCYJSYY[PGKBZGY[YWJKGKLJYWKPJ
QHY[W[DZLSGMRYPYWWCCKZNKYYGTTNJJNYKKZYTCJNMCYLQLYPYQFQRPZSLWBTGKJFYX
JWZLTBNCXJJJJTXDTTSQZYCDXXHGCK[PHFFSS[YBGXLPPBYLL[HLXS[ZM[JHSOJNGHDZ
QYKLGJHSGQZHXQGKEZZWYSCSCJXYEYXADZPMDSSMZJZQJYZC[J[WQJBYZPXGZNZCPWHK
XHQKMWFBPBYDTJZZKQHY"

"LYGXFPTYJYYZPSZLFCHMQSHGMXXSXJ[[DCSBBQBEFSJYHXWGZKPYLQBGLDLCCTNMAYD
DKSSNGYCSGXLYZAYBNPTSDKDYLHGYMYLCXPY[JNDQJWXQXFYYFJLEJPZRXCCQWQQSBNK
YMGPLBMJRQCFLNYMYQMSQYRBCJTHZTQFRXQHXMJJCJLXQGJMSHZKBSWYEMYLTXFSYDSW
LYCJQXSJNQBSCTYHBFTDCYZDJWYGHQFRXWCKQKXEBPTLPXJZSRMEBWHJLBJSLYYSMDXL
CLQKXLHXJRZJMFQHXHWY"

"WSBHTRXXGLHQHFNM[YKLDYXZPYLGG[MTCFPAJJZYLJTYANJGBJPLQGDZYQYAXBKYSEC
JSZNSLYZHSXLZCGHPXZHZNYTDSBCJKDLZAYFMYDLEBBGQYZKXGLDNDNYSKJSHDLYXBCG
HXYPKDJMMZNGMMCLGWZSZXZJFZNMLZZTHCSYDBDLLSCDDNLKJYKJSYCJLKWHQASDKNHC
SGANHDAASHTCPLCPQYBSDMPJLPZJOQLCDHJJYSPRCHN[NNLHLYYQYHWZPTCZGWWMZFFJ
QQQQYXACLBHKDJXDGMMY"

"DJXZLLSYGXGKJRYWZWYCLZMSSJZLDBYD[FCXYHLXCHYZJQ[[QAGMNYXPFRKSSBJLYXY
SYGLNSCMHZWWMNZJJLXXHCHSY[[TTXRYCYXBYHCSMXJSZNPWGPXXTAYBGAJCXLY[DCCW
ZOCWKCCSBNHCPDYZNFCYYTYCKXKYBSQKKYTQQXFCWCHCYKELZQBSQYJQCCLMTHSYWHMK
TLKJLYCXWHEQQHTQH[PQ[QSCFYMNDMGBWHWLGSLLYSDLMLXPTHMJHWLJZYHZJXHTXJLH
XRSWLWZJCBXMHZQXSDZP"

"MGFCSGLSXYMJSHXPJXWMYQKSMYPLRTHBXFTPMHYXLCHLHLZYLXGSSSSTCLSLDCLRPBH
ZHXYYFHB[GDMYCNQQWLQHJJ[YWJZYEJJDHPBLQXTQKWHLCHQXAGTLXLJXMSL[HTZKZJE
CXJCJNMFBY[SFYWYBJZGNYSDZSQYRSLJPCLPWXSDWEJBJCBCNAYTWGMPAPCLYQPCLZXS
BNMSGGFNZJJBZSFZYNDXHPLQKZCZWALSBCCJX[YZGWKYPSGXFZFCDKHJGXDLQFSGDSLQ
WZKXTMHSBGZMJZRGLYJB"

"PMLMSXLZJQQHZYJCZYDJWBMYKLDDPMJEGXYHYLXHLQYQHKYCWCJMYYXNATJHYCCXZPC
QLBZWWYTWBQCMLPMYRJCCCXFPZNZZLJPLXXYZTZLGDLDCKLYRZZGQTGJHHGJLJAXFGFJ
ZSLCFDQZLCLGJDJCSNZLLJPJQDCCLCJXMYZFTSXGCGSBRZXJQQCTZHGYQTJQQLZXJYLY
LBCYAMCSTYLPDJBYREGKLZYZHLYSZQLZNWCZCLLWJQJJJKDGJZOLBBZPPGLGHTGZXYGH
ZMYCNQSYCYHBHGXKAMTX"

"YXNBSKYZZGJZLQJDFCJXDYGJQJJPMGWGJJJPKQSBGBMMCJSSCLPQPDXCDYYKY[CJDDY
YGYWRHJRTGZNYQLDKLJSZZGZQZJGDYKSHPZMTLCPWNJAFYZDJCNMWESCYGLBTZCGMSSL
LYXQSXSBSJSBBSGGHFJLYPMZJNLYYWDQSHZXTYYWHMZYHYWDBXBTLMSYYYFSXJC[DXXL
HJHF[SXZQHFZMZCZTQCXZXRTTDJHNNYZQQMNQDMMG[YDXMJGDHCDYZBFFALLZTDLTFXM
XQZDNGWQDBDCZJDXBZGS"

"QQDDJCMBKZFFXMKDMDSYYSZCMLJDSYNSBRSKMKMPCKLGDBQTFZSWTFGGLYPLLJZHGJ[
GYPZLTCSMCNBTJBQFKTHBYZGKPBBYMTDSSXTBNPDKLEYCJNYDDYKZDDHQHSDZSCTARLL
TKZLGECLLKJLQJAQNBDKKGHPJTZQKSECSHALQFMMGJNLYJBBTMLYZXDCJPLDLPCQDHZY
CBZSCZBZMSLJFLKRZJSNFRGJHXPDHYJYBZGDLQCSEZGXLBLGYXTWMABCHECMWYJYZLLJ
JYHLG[DJLSLYGKDZPZXJ"

"YYZLWCXSZFGWYYDLYHCLJSCMBJHBLYZLYCBLYDPDQYSXQZBYTDKYXJY[CNRJMPDJGKL
CLJBCTBJDDBBLBLCZQRPPXJCJLZCSHLTOLJNMDDDLNGKAQHQHJGYKHEZNMSHRP[QQJCH
GMFPRXHJGDYCHGHLYRZQLCYQJNZSQTKQJYMSZSWLCFQQQXYFGGYPTQWLMCRNFKKFSYYL
QBMQAMMMYXCTPSHCPTXXZZSMPHPSHMCLMLDQFYQXSZYYDYJZZHQPDSZGLSTJBCKBXYQZ
JSGPSXQZQZRQTBDKYXZK"
```

```
"HHGFLBCSMDLDGDZDBLZYYCXNNCSYBZBFGLZZXSWMSCCMQNJQSBDQSJTXXMBLTXZCLZS
HZCXRQJGJYLXZFJPHYMZQQYDFQJJLZZNZJCDGZYGCTXMZYSCTLKPHTXHTLBJXJLXSCDQ
XCBBTJFQZFSLTJBTKQBXXJJLJCHCZDBZJDCZJDCPRNPQCJPFCZLCLZXZDMXMPHJSGZGS
ZZQLYLWTJPFSYASMCJBTZKYCWMYTCSJJLJCQLWZMALBXYFBPNLSFHTGJWEJJXXGLLJST
GSHJQLZFKCGNNNSZFDEQ"

"FHBSAQTGYLBXMMYGSZLDYDQMJJRGBJTKGDHGKBLQKBDMBYLXWCXYTTYBKMRTJZXQJBH
LMHMJJZMQASLDCYXYQDLQCAFYWYXQHZ";
const char* cName = name.c_str();
std::string result;
int H = 0, L = 0, W = 0, j = 0;
size_t stringlen = ::strlen(cName);
for (int i = 0; i < stringlen; ++i) {
H = (unsigned char)(cName[i + 0]);
L = (unsigned char)(cName[i + 1]);
if (H < 0xA1 || L < 0xA1) {
    result += cName[i];
    continue;
}
W = (H - 160) * 100 + L - 160;
if (W > 1600 && W < 5590) {
    bool has = false;
    for (j = 22; j >= 0; j--) {
        if (W >= secPosValue[j]) {
            result += initials[j];
            i++;
            has = true;
            break;
        }
    }
    continue;
}
i++;
W = (H - 160 - 56) * 94 + L - 161;
if (W >= 0 && W <= 3007)
    result += secondSecTable[W];
else {
    result += (unsigned char)H;
    result += (unsigned char)L;
}
}
return result;
}
```

1.5 关键字高亮显示

```
// 颜色高亮显示一段字符串
static void ColourPrintf(const char* str)
{
    // 0-黑 1-蓝 2-绿 3-浅绿 4-红 5-紫 6-黄 7-白 8-灰 9-淡蓝 10-淡绿
    // 11-淡浅绿  12-淡红 13-淡紫 14-淡黄 15-亮白
    //颜色：前景色 + 背景色*0x10
    //例如：字是红色，背景色是白色，即 红色 + 亮白 =  4 + 15*0x10
WORD color = 4 + 15 * 0x10;
WORD colorOld;
HANDLE handle = ::GetStdHandle(STD_OUTPUT_HANDLE);
```

```
CONSOLE_SCREEN_BUFFER_INFO csbi;
GetConsoleScreenBufferInfo(handle, &csbi);
colorOld = csbi.wAttributes;
SetConsoleTextAttribute(handle, color);
printf("%s", str);
SetConsoleTextAttribute(handle, colorOld);
}

////////////////////////////////////////////////////////////////////////
////////

//高亮显示核心: 字符串的分割
void DataManager::SplitHighlight(const string& str, const string&
key, string& prefix, string& highlight, string& suffix)
{
    string strlower(str), keylower(key);
    transform(str.begin(), str.end(), strlower.begin(), ::tolower);
    transform(key.begin(), key.end(), keylower.begin(), ::tolower);

    // 1.如果是中文搜索，直接可以匹配strlower中的子串，则直接分离
    size_t pos = strlower.find(keylower);
    if (pos != std::string::npos)
    {
        prefix = str.substr(0, pos);
        highlight = str.substr(pos, keylower.size());
        suffix = str.substr(pos + keylower.size(), string::npos);
        return;
    }

    // 2.如果是拼音全拼搜索，则需要匹配分离子串汉字和拼音全拼
    std::string key_pinyin = ChineseConvertPinYinAllSpell(keylower);
    std::string str_pinyin = ChineseConvertPinYinAllSpell(strlower);
    pos = str_pinyin.find(key_pinyin);
    if (pos != std::string::npos)
    {
        size_t str_index = 0;
        size_t pinyin_index = 0;

        size_t highlight_index = 0;
        size_t highlight_len = 0;

        while (str_index < strlower.size())
        {
            // 如果拼音位置已经走到子串匹配的位置，则原串的位置就是高亮的起始位置
            if (pinyin_index == pos)
            {
                highlight_index = str_index;
            }

            // 如果拼音位置已经走完keylower的位置，则原串的位置就是高亮的结束位
置
            if (pinyin_index >= pos + key_pinyin.size())
            {
                highlight_len = str_index - highlight_index + 1;
                break;
            }

```

```
            if (strlower[str_index] >= 0 && strlower[str_index] <= 127)
            {
                // 如果是字符则各跳一个位置
                ++str_index;
                ++pinyin_index;
            }
            else
            {
                // 如果是汉字原串的汉字跳两个字节(gbk的汉字是两个字符)，拼音跳一个全拼的位置
                std::string substr(strlower, str_index, 2);
                std::string subpinyin = ChineseConvertPinYinAllSpell(substr);

                str_index += 2;
                pinyin_index += subpinyin.size();
            }
        }

        prefix = str.substr(0, highlight_index);
        highlight = str.substr(highlight_index, highlight_len);
        suffix = str.substr(highlight_index + highlight_len, std::string::npos);

        return;
    }


    // 3.如果是拼音首字母，则需要匹配首字母和汉字
    std::string key_initials = ChineseConvertPinYinInitials(keylower);
    std::string str_initials = ChineseConvertPinYinInitials(strlower);
    pos = str_initials.find(key_initials);
    if (pos != std::string::npos)
    {
        size_t str_index = 0;
        size_t initials_index = 0;

        size_t highlight_index = 0;
        size_t highlight_len = 0;

        while (str_index < strlower.size())
        {
            // 如果拼音位置已经走到子串匹配的位置，则原串的位置就是高亮的起始位置
            if (initials_index == pos)
            {
                highlight_index = str_index;
            }

            // 如果拼音位置已经走完keylower的位置，则原串的位置就是高亮的结束位置
            if (initials_index >= pos + key_initials.size())
            {
                highlight_len = str_index - highlight_index + 1;
                break;
            }
```

```
            if (strlower[str_index] >= 0 && strlower[str_index] <= 127)
            {
                // 如果是字符则各跳一个位置
                ++str_index;
                ++initials_index;
            }
            else
            {
                // 如果是汉字原串的汉字跳两个字节(gbk的汉字是两个字符)，首字母跳一个
                str_index += 2;
                ++initials_index;
            }
        }

        prefix = str.substr(0, highlight_index);
        highlight = str.substr(highlight_index, highlight_len);
        suffix = str.substr(highlight_index + highlight_len, std::string::npos);
        return;
    }

    // 如果走到最后没匹配高亮，则按不高亮处理
    prefix = strlower;
    highlight.clear();
    suffix.clear();
}
```

2.扫描模块的实现

```
class ScanManager
{
public:
    //static ScanManager& CreateInstance(const std::string& path = "D:\\");
    static ScanManager& CreateInstance(const std::string& path = "C:\\");
    void ScanDirectory(const std::string& path);
    void StartScan(const std::string& path);
private:
    ScanManager();
    std::vector<std::string> _entry_dirs; // 扫描的入口目录
};
////////////////////////////////////////////////////////////////////////
void ScanManager::StartScan(const std::string& path)
{
    cout << "扫描线程开始:" << path << endl;

    _entry_dirs.push_back(path);
    while (1)
    {
        std::this_thread::sleep_for(std::chrono::seconds(5));

        for (const auto& dir : _entry_dirs)
```

```
        {
            ScanDirectory(dir);
        }
    }
}

ScanManager& ScanManager::CreateInstance(const std::string& path)
{
    static ScanManager inst;
    static std::thread scan_thread(&StartScan, &inst, path);
    scan_thread.detach();

    return inst;
}

```

3.数据管理模块的实现

3.1 数据库封装

```
//////////////////////////////////////////////////////////
// SqliteManager是对Sqlite的接口进行一层简单的封装。
class SqliteManager
{
public:
    SqliteManager()
        :_db(nullptr)
    {}
    ~SqliteManager()
    {
        Close();
    }
    void Open(const std::string& path);
    void Close();
    void ExecuteSql(const std::string& sql);
    void GetTable(const std::string& sql, int& row, int& col, char**&
ppRet);

    SqliteManager(const SqliteManager&) = delete;
    SqliteManager& operator=(const SqliteManager&) = delete;
private:
    sqlite3* _db;       // 数据库对象
};
```

3.2 自动获取数据表机制

```
// RAII(资源的获取及初始化)，自动释放sqlite返回的二维数组
class AutoGetTable
{
public:
    AutoGetTable(SqliteManager* dbObject, const std::string& sql, int&
row, int& col, char**& ppRet)
        : _dbObject(dbObject)
        , _ppObject(0)
    {
        _dbObject->GetTable(sql, row, col, ppRet);
        _ppObject = ppRet;
```

```
    }
    virtual ~AutoGetTable()
    {
        if (_ppObject)
            sqlite3_free_table(_ppObject);
    }
private:
    AutoGetTable(const AutoGetTable&) = delete;
    AutoGetTable& operator=(const AutoGetTable&) = delete;
private:
    SqliteManager*      _dbObject;
    char**              _ppObject;
};
```

3.3 数据管理

```
//////////////////////////////////////////////////////////////////
// DataManager提供数据库管理的接口
#define DOC_TABLE "tb_doc"
#define DOC_DB "doc.db"
class DataManager
{
public:
    ~DataManager()
    {
        _dbmgr.Close();
    }
    static DataManager& DataManager::GetInstance()
    {
        static DataManager inst;
        return inst;
    }
    void Init();
    void GetDocs(const std::string path, std::set<std::string>& docs);
    void InsertDoc(const std::string path, std::string doc);
    void DeleteDoc(const std::string path, std::string doc);
    void Search(const std::string& key,
std::vector<std::pair<std::string, std::string>>& doc_paths);
    static void SplitHighlight(const std::string& str, const
std::string& key,
        std::string& prefix, std::string& highlight, std::string&
suffix);
private:
    DataManager()
    {
        // 打开数据库
        _dbmgr.Open(DOC_DB);
        Init();
    }
    DataManager(const DataManager&) = delete;
    SqliteManager   _dbmgr;
    std::mutex      _mtx;
};
```

5.系统/代码重构

5.1系统重构

1、扫描模块的单例化以及线程化
为什么搜索工具需要进行单列化:整个工具在搜索文档的过程中，其扫描实例只需要一个，这个扫描实例只需要把需要扫描的路径下面的文档(包括子目录)全部扫进数据库即可，不需要多个扫描对象，如果有多个可能会造成数据库数据重复，或者产生脏数据，从而导致搜索工具的不准确。
为什么搜索工具需要进行线程化:整个搜索过程中，有可能本地文档会有增加或删除的情况，如果扫描实例只扫描一次就停止了扫描，会导致本地数据与数据库的数据不对称，因此，需要时刻扫描本地数据，从而达到对数据库文件的实时更新（本地的实时理论上也是有一定的时间间隔，或称为延迟），一旦线程化，我们可以通过子线程来完成扫描工作，而主线程继续完成与用户的交互，因此，需要线程化。

```
// 多线程基础知识
#include<thread>
thread th(thread_fun, thread_fun_arg); //线程实例化
th.detach();//线程分离
th.join(); //线程等待
//线程休眠时间
#include<chrono>--用于时间延时 获取时间之类的
//下面这里返回一个毫秒级别的时间间隔参数值，间隔10毫秒
std::chrono::milliseconds(10)
this_thread::sleep_for（）就是让此线程休眠，可以传入休眠的时间
this_thread::sleep_for(std::chrono::milliseconds(10));让本线程休眠10毫秒
```

2、数据管理模块的单例化
针对不同的搜索，整个过程用到的数据库实例为同一个，针对管理数据库实例的对象，数据管理对象，此时也需要保证不同的搜索程序，使用的是同一个数据管理对象，因此单例化能够完成任务。

5.2代码重构

```
1、添加必要代码注释
2、必要位置添加日志
```

6.项目打包发布

```
1、工具
InstallShield工具对代码程序打包发布
```

7.项目扩展

```
1、数据库、表名、搜索方式等 支持可配置化
2、提供搜索显示界面
```

8.项目总结

```
1、项目穿插讲解
2、针对面试关注点解析
```

9.界面处理

system 函数

函数名	system
功 能	发出一个 DOS 命令
用 法	int system(char *command)
描述	system 函数已经被收录在标准 c 库中，可以直接调用
功能使用	
设置 cmd 窗口标题	system("title title_name")
设置窗口宽度高度	system("mode con cols=m lines=n")
设置窗体颜色	system("color XY") 其中 color 后面的 X 是背景色代号，Y 是前景色代号 0=黑色 1=蓝色 2=绿色 3=湖蓝色 4=红色 5=紫色 6=黄色 7=白色 8=灰色 9=淡蓝色 A=淡绿色 B=淡浅绿色 C=淡红色 D=淡紫色 E=淡黄色 F=亮白色
系统暂停	system("pause")
清 屏	system("cls")

控制台光标位置设置

1、引入头文件 window.h

2、获取控制台句柄 HANDLE handle = GetStdHandle(STD_OUTPUT_HANDLE);

3、设置光标位置

```
SetConsoleCursorPosition(handle, pos);
其中:BOOL SetConsoleCursorPosition(
HANDLE hConsoleOutput,  // handle to console screen buffer
COORD dwCursorPosition ); //new cursor position coordinates
typedef struct COORD
{
      SHORT X;      // horizontal coordinate
      SHORT Y;      // vertical coordinate
}COORD;
```

控制台光标隐藏设置

1、引入头文件 window.h

2、定义光标信息结构体变量 CONSOLE_CURSOR_INFO cursor_info = {1, 0};

```
typedef struct _CONSOLE_CURSOR_INFO {
DWORD dwSize; // 光标百分比大小
BOOL bVisible; // 是否可见
} CONSOLE_CURSOR_INFO, *PCONSOLE_CURSOR_INFO;
```

3、获取控制台句柄 HANDLE handle = GetStdHandle(STD_OUTPUT_HANDLE);

4、调用设置控制台光标信息函数 SetConsoleCursorInfo(handle, & cursor_info);

```
其中:BOOL SetConsoleCursorInfo( // 设置光标信息
      HANDLE hConsoleOutput, //句柄
```

CONST CONSOLE_CURSOR_INFO *lpConsoleCursorInfo // 光标信息);

```
#ifndef _SYSFRAME_H_
#define _SYSFRAME_H_

#include"Common.h"

#define MAX_TITLE_SIZE 100

void S nt x, int y);
void HideCursor();
void DrawCol(int x, int y);
void DrawRow(int x, int y);

void DrawFrame(char *title);
void DrawMenu();
void SystemEnd();


#endif /* _SYSFRAME_H_ */

////////////////////////////////////////////////////////////////////////
////
#include"sysframe.h"

#define WIDTH 120
#define HEIGHT  30

void SetCurPos(int x, int y)
{
    HANDLE handle = GetStdHandle(STD_OUTPUT_HANDLE);
    COORD pos = {x, y};
    SetConsoleCursorPosition(handle, pos);
}

void HideCursor()
{
    CONSOLE_CURSOR_INFO  cursor_info = {100, 0};
    HANDLE handle = GetStdHandle(STD_OUTPUT_HANDLE);
    SetConsoleCursorInfo(handle, &cursor_info);
}

void DrawCol(int x, int y)
{
    int i;
    for(i=0; i<HEIGHT; ++i)
    {
        SetCurPos(x, y+i);
        printf("||");
    }
}

void DrawRow(int x, int y)
{
    int i;
    for(i=0; i<WIDTH-4; ++i)
    {
```

```
        SetCurPos(x+i, y);
        printf("=");
    }
}

void DrawFrame(char *title)
{
    char buffer[MAX_TITLE_SIZE+6+1] = "title "; //6:title%20  1:\0
    strcat(buffer, title);
    system(buffer); //设置系统标题

    char mode[128] = {0};
    sprintf(mode, "mode con cols=%d lines=%d",  WIDTH,HEIGHT);

    system(mode); //设置控制台的长度和宽度
    system("color 0F");//设置颜色

    DrawCol(0, 0);
    DrawCol(WIDTH-2, 0);
    DrawRow(2, 0);
    DrawRow(2, 2);
    DrawRow(2, 4);
    DrawRow(2, HEIGHT-4);
    DrawRow(2, HEIGHT-2);
}

void SystemEnd()
{
    SetCurPos(0, HEIGHT-1);
}

extern char *title;
void DrawMenu()
{
    //根据自己的需求进行实现
    //标题的设置
    SetCurPos((WIDTH-4-strlen(title))/2, 1);
    printf("%s", title);

    //名称  路径
    SetCurPos(2, 3);
    printf("%-30s %-85s","名称","路径");

    //退出设置
    SetCurPos((WIDTH-4-strlen("exit 退出系统 ."))/2 ,HEIGHT-3);
    printf("%s", "exit 退出系统 .");

    DrawRow(2, HEIGHT-6);
    //SetCurPos((WIDTH-4-strlen("请输入:>"))/2, 15);

    SetCurPos(2, HEIGHT-5);
    printf("%s", "请输入:>");
}
```

10、扩展 everything实现原理

本地文件搜索工具 Everything 为什么速度这么快?

Everything并不扫描整个磁盘，只是读取磁盘上的USN日志，并建立索引，所以速度飞快。
但因此缺点也明显：
1、只支持NTFS格式的分区，因为USN日志是NTFS专有的。在FAT、FAT32格式分区上无法使用Everything。
2、只索引文件名称、日期和大小，不索引文件内容和附加属性。
3.由于Everything只读取USN日志，所以也无法搜索网络邻居及映射的网络文件夹。

一、USN Journal及MFT原理
everything搜索文件的速度之所以快得令人愤怒，主要原因是利用了NTFS的USNJournal特性，直接从系统的主文件表里读取文件信息。
USNJournal（Update SequenceNumber Journal）也被称作Change Journal。微软在NTFS 5.0（Windows 2000）中引入了USN Journal特性。USN Journal实际上是NTFS分区的一个日志文件，包含了所有对分区操作（文件操作）的记录。操作系统为每一个扇区单独维护一个USN Journal。当扇区的文件有变化时，操作系统会往USN Journal文件中追加一条记录，该记录包含文件名、变化发生的时间、变化的原因等信息，而不包含变化的内容。每一条记录用一个64位数字标识，称作USN（UpdateSequence Number）。微软用每一条记录在日志文件中的偏移作为该记录的USN，这样可以快速地通过USN获取到对应的记录。显而易见，USN是递增的，但是不连续。同时，由于文件名有长有短，每条记录的长度也不固定。在日志文件中，以文件块的形式存储记录。每个文件块大小为4K（USN_PAGE_SIZE），按每条记录100字节计，可以存储30~40条记录。单条记录不允许跨块存储。
USNJournal也有一个64位的ID（Journal ID）。当文件/目录发生变化时，系统除了往USN Journal追加一条日志外，也会更新Journal ID。如果Journal ID没有更新，意味着文件系统没有任何变化。
对NTFS分区，文件信息存储于主文件表（Master File Table，MFT）。MFT是NTFS的核心。主文件表里的每条记录包含文件和目录的所有信息，例如文件名/目录名、路径、大小、属性等。此外，还会存储该文件/目录最后一次变化对应的USN，系统在更新USN Journal时，也会更新这个字段。
小提示：可以用FSUtil.exe获取NTFS信息。FSUtil.exe FSInfoNTFSInfo C:
二、相关API
Windows提供了一系列API供应用程序访问USN Journal及MFT。相关的函数及数据结构罗列如下：
l DeviceIoControl
l FSCTL_QUERY_USN_JOURNAL
l struct USN_JOURNAL_DATA
l FSCTL_ENUM_USN_DATA
l FSCTL_READ_USN_JOURNAL
l struct USN_RECORD

DeviceIoControl如何使用请查阅MSDN。FSCTL_QUERY_USN_JOURNAL和FSCTL_READ_USN_JOURNAL是DeviceIoControl的两个控制码，分别用于读取USN Journal统计信息和USN Journal记录。

三、实现思路
出于项目保密考虑，这里只大致介绍一下实现思路。
首先对每一个分区用FSCTL_ENUM_USN_DATA遍历文件和目录，建立索引；然后用FSCTL_READ_USN_JOURNAL监控系统中的文件变化（可只监听感兴趣的变化，如新建、删除、改名等操作），更新索引；退出程序时将索引保存到文件中，供下次启动时加载。

四、扩展思考

1. 如何由FRN获取文件/目录的路径?
   Windows没有提供相应的API来完成这个功能。只有先建立索引树，然后遍历索引树得到完整的路径。

2. 关闭everything之后的文件变化，重启everything之后如何扫描到？
   用FSCTL_ENUM_USN_DATA遍历至MFT的最后一项时，会返回USN Journal最后一项的USN。保存下来，下次重启Everything时，从该USN重新开始遍历即可。
3. 怎样从MFT里获取文件的访问时间？

利用控制码FSCTL_GET_NTFS_FILE_RECORD可以获取MFT里的一条记录。使用时需要提供文件的FRN（FileReferenceNumber）。实现细节可参考以下文章
http://www.codeproject.com/Articles/9293/Undelete-a-file-in-NTFS
http://blog.csdn.net/problc/article/details/5971825

——————————————————