拓展

1. 多文件支持
2. 支持正则表达
3. 模糊查询
4. 可视化搜索结果，如模仿编译器报错的方式
5. 新的哈希函数
6. 支持多语言
7. 性能优化与预处理
8. 结果排序
9. 用户界面
10. 持久化存储
11. 日志功能，记录查询历史等
12. 支持更多的文件格式，pdf，word等

TextSearchEngine/

│

├── main.cpp

├── TextSearchEngine.h

├── TextSearchEngine.cpp

├── RegexSearch.h

└── RegexSearch.cpp

已实现：

用户选择查询类型

无限查询+退出选项

#include "TextSearchEngine.h"

#include "RegexSearch.h"

#include <iostream>

#include <limits> // 新增

int main() {

    TextSearchEngine engine;

    // 加载 source-text 文件夹中的多个文件

    std::vector<std::string> files = {

        "source-text/text1.txt",

        "source-text/text2.txt"

    };

    engine.loadTexts(files);

    std::cout << "欢迎使用文本搜索引擎！\n";

    std::cout << "功能说明：\n1. 精确查询:输入为一个句子,匹配跟这个句子完全一样的句子\n2. 包含查询:输入为一个单词或词组,输出该词组所在的句子和位置\n3. 正则查询:按照正则表达式规则进行匹配\n4. 退出\n";

    int choice;

    bool running = true;

    while (running) {

        int choice;

        bool validInput;

        do {

            std::cout << "请选择查询类型:\n1. 精确查询\n2. 包含查询\n3. 正则查询\n4. 退出\n输入选项:\n";

            std::cin >> choice;

            if (std::cin.fail()) {

                std::cin.clear();

                std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

                validInput = false;

            } else {

                validInput = (choice >= 1 && choice <= 4);

            }

            if (!validInput) {

                std::cout << "无效选项，请重试。";

            }

        } while (!validInput);

        std::string input;

        switch (choice) {

            case 1:

                std::cout << "输入关键词: ";

                std::cin >> input;

                engine.exactQuery(input);

                break;

            case 2:

                std::cout << "输入关键词: ";

                std::cin >> input;

                engine.fuzzyQuery(input);

                break;

            case 3:

                std::cout << "输入正则表达式: ";

                std::cin.ignore();

                std::getline(std::cin, input);

                RegexSearch::regexQuery(engine.getSentences(), input);

                break;

            case 4:

                running = false;

                break;

            default:

                std::cout << "无效选项，请重试。";

        }

    }

    return 0;

}