Project3 实验报告

22336216　陶宇卓

1、　程序功能简要说明。

文本信息存储于一个文本文件中。

1. 待统计的词汇集合一次输入完毕，即统计工作必须在程序运行一次之后，就全部完成。程序的基本输出结果，要求是每个词的出现次数（降序排列），以及出现位置所在的行号。
2. 输出所有出现单词的出现次数以及行号。

程序可能还会有漏洞，但是大部分文本都可以识别。

2、　程序运行截图，包括计算功能演示、部分实际运行结果展示、命令行或交互式界面效果等。

接下来以我自定义的文本为例

文本内容：

The Wonders of Computers

In today's digital age, computers have become an indispensable part of our lives, revolutionizing the way we work, communicate, and live.

These marvels of technology have transformed the world into a global village, connecting people from different corners of the Earth at the click of a button.

Computers are incredibly versatile machines.

They can process vast amounts of data at lightning speed, making complex calculations and solving intricate problems with ease.

From the smallest smartphones to the most powerful supercomputers, these electronic devices have reshaped industries, enabling advancements in fields such as medicine, engineering, and scientific research.

One of the most remarkable features of computers is their ability to store and retrieve information.

With the advent of the internet, a vast ocean of knowledge is accessible to anyone with a computer and an internet connection.

This instant access to information has transformed education, allowing students and scholars to explore the depths of human understanding from the comfort of their homes.

Moreover, computers have revolutionized communication.

Emails, social media platforms, and instant messaging apps have made it possible for people to stay connected regardless of geographical distances.

Video conferencing tools have transformed the way businesses operate, enabling virtual meetings and collaborations across continents.

In the world of creativity, computers have opened up new avenues for artists, musicians, and designers.

Digital art, music production software, and graphic design tools have unleashed boundless creativity, allowing individuals to express themselves in ways previously unimaginable.

However, with great power comes great responsibility.

As we embrace the wonders of computers, it's crucial to be aware of cybersecurity threats and the importance of safeguarding our digital identities.

Understanding the ethical implications of technology and promoting responsible computer use are essential steps toward creating a safer online environment for everyone.

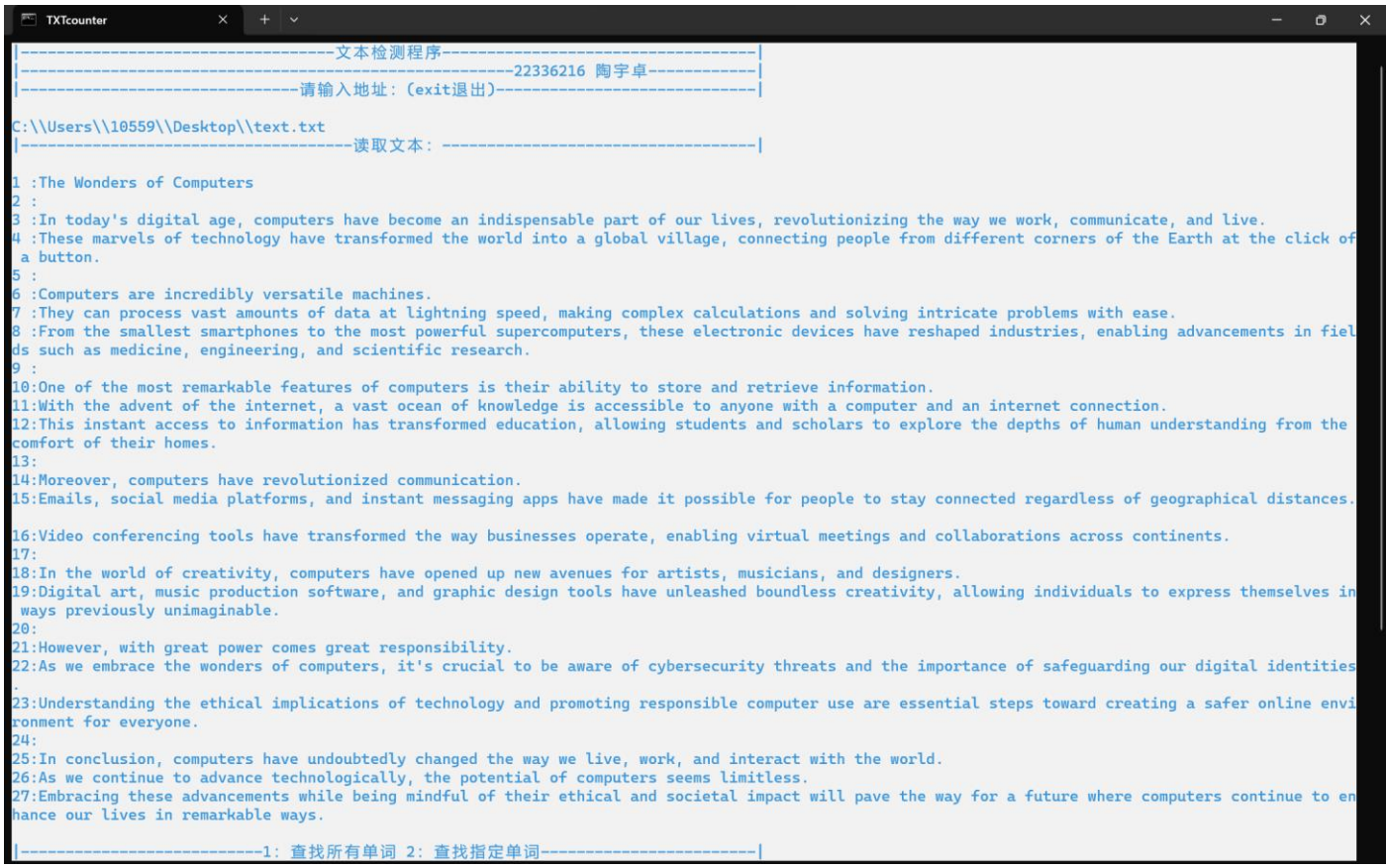
In conclusion, computers have undoubtedly changed the way we live, work, and interact with the world.

As we continue to advance technologically, the potential of computers seems limitless.

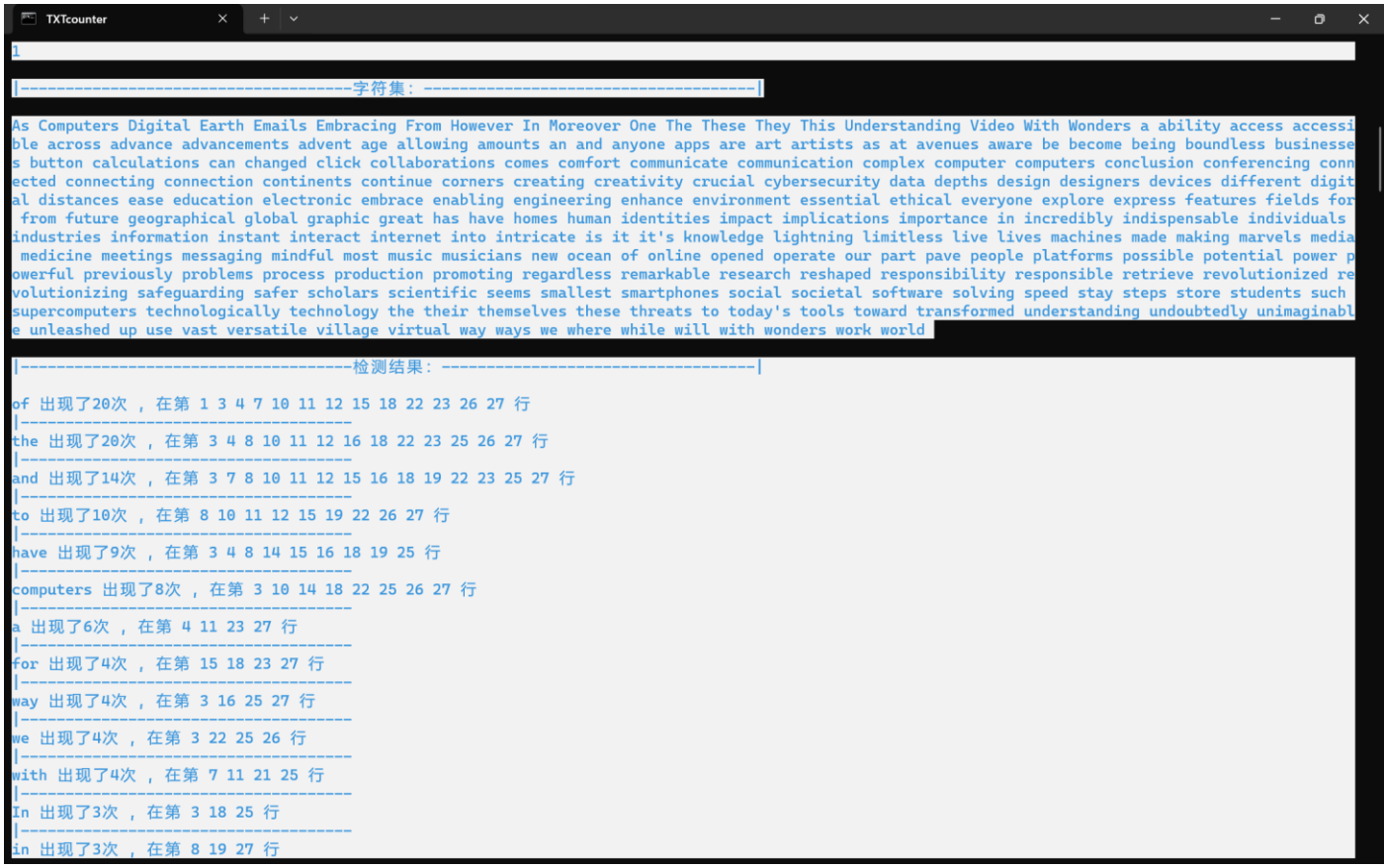
Embracing these advancements while being mindful of their ethical and societal impact will pave the way for a future where computers continue to enhance our lives in remarkable ways.

初始界面：

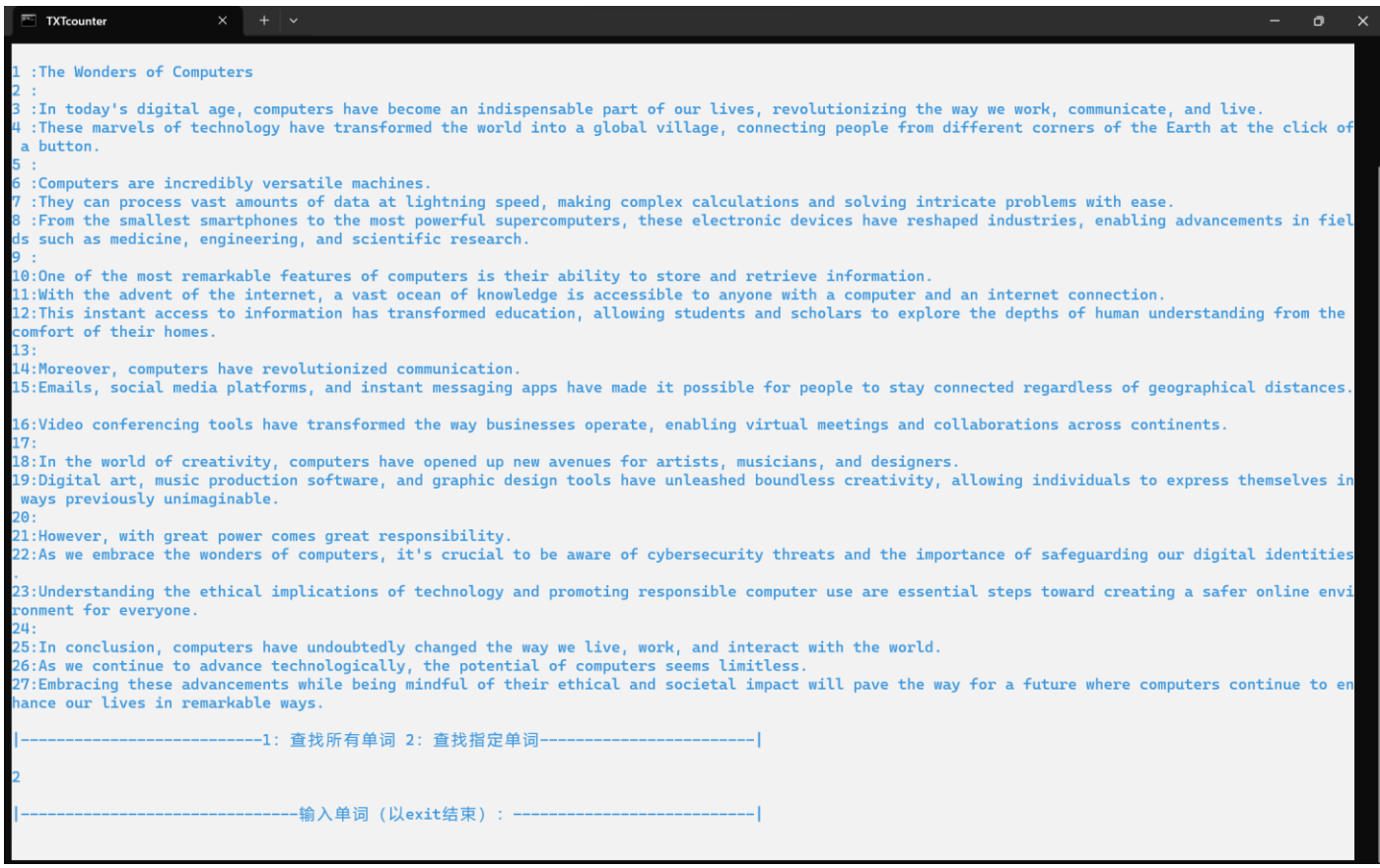
输入地址 C:\\Users\\10559\\Desktop\\text.txt：

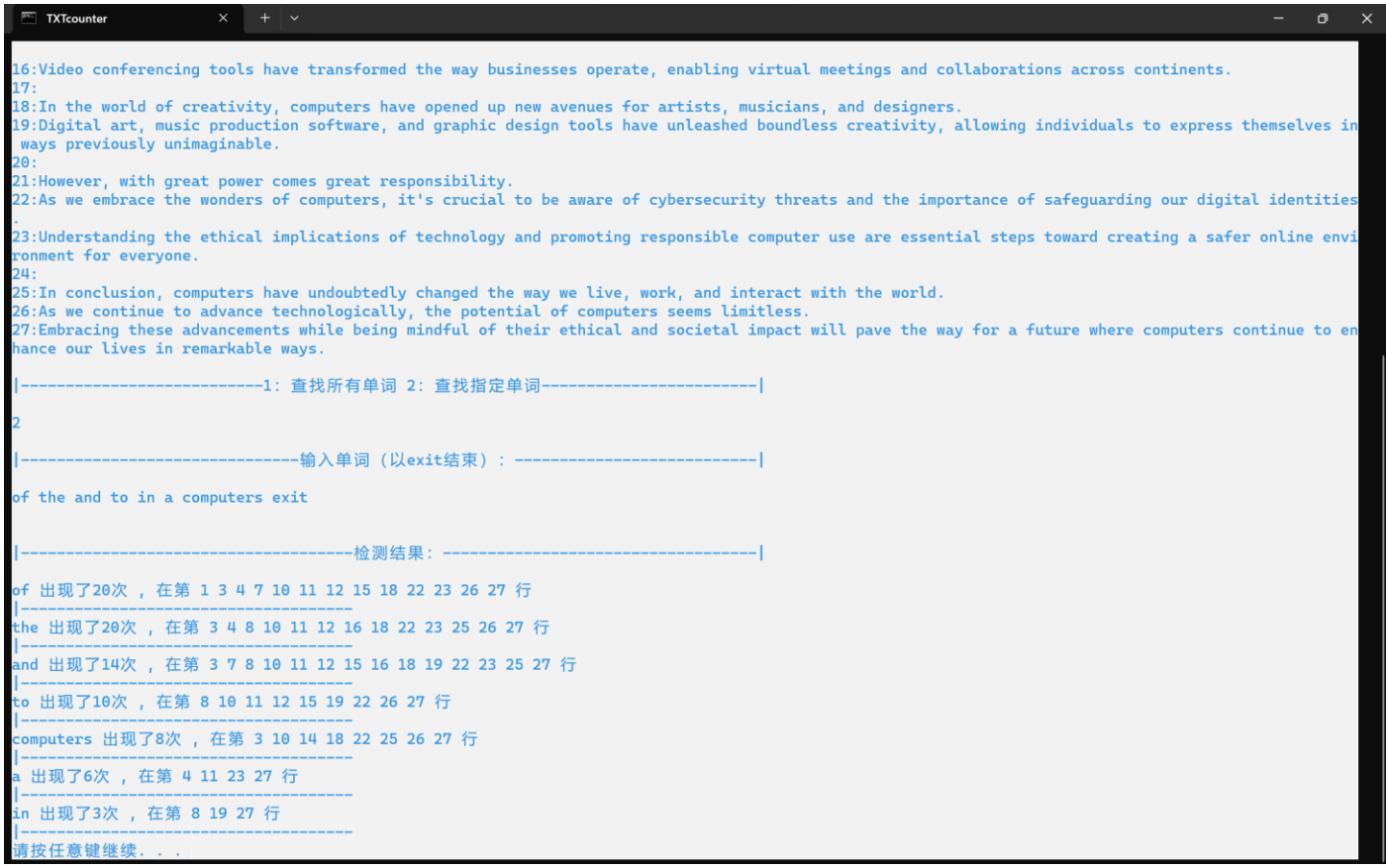


输入 1：



同理按 2，输入单词 of the and to in a computers exit 显示：





3、 部分关键代码及其说明。

自定义 Compare 类：包含一个操作符重载函数，为了在排序时能根据特定的 value 值排序：

```
class Compare {
public:
    bool operator()(const pair<string, pair<int, string>>& a, const pair<string, pair<int, string>>& b) {
        if (a.second.first != b.second.first) {
            return a.second.first > b.second.first;
        }
        else return a.first < b.first;
    }
};
```

三个容器：全部字符集 `vector<string> keyword`，给定字符集 `vector<string> targetword`，统计结果 `map<string, pair<int, string>> word_count`。

```
vector<string> keyword;
vector<string> targetword;
map<string, pair<int, string>> word_count;
```

计算行数以及读取特定行：根据所输入的文本地址来遍历得到行数，并访问得到一行字符串：

```cpp
int CountLines(const char* filename) {
    ifstream ReadFile;
    int count = 0;
    string tmp;
    ReadFile.open(filename, ios::in);
    if (ReadFile.fail()) {
        return 0;
    }
    else {
        while (getline(ReadFile, tmp, '\n')) {
            count++;
        }
        ReadFile.close();
        return count;
    }
}

string Readline(const char* filename, int line) {
    fstream file;
    string tmp;
    file.open(filename, ios::in);
    int i = 0;
    while (getline(file, tmp) && i < line - 1) {
        i++;
    }
    file.close();
    return tmp;
}
```

获取全部字符集：使用 istringstream 遍历一行，去掉符号，得到 keyword

```cpp
void GetKeyWord(string tmp) {
    istringstream ss(tmp);
    string word;
    while (ss >> word) {
        while (ispunct(word[word.size() - 1]) && word.size()>1) {
            word = word.substr(0, word.size() - 1);
        }
        while (ispunct(word[0]) && word.size() > 1) {
            word = word.substr(1, word.size() - 1);
        }
        if (ispunct(word[0])) continue;
        if (find(keyword.begin(), keyword.end(), word) == keyword.end()) keyword.push_back(word);
        else;
    }
    return;
}
```

KMP 算法：

```
int KMP(string line, string target) {
    int ans = 0;
    int i = 0, j = 0;
    int lineLen = line.size(), targetLen = target.size();
    while (i <= lineLen) {
        if (j == targetLen) {
            if (isalnum(line[i])) {
                j = 0;
                goto count;
            }
            else if (i - j - 1 >= 0) {
                if (isalnum(line[i - j - 1])) {
                    j = 0;
                    goto count;
                }
            }
            j = 0;
            ans++;
        }
        if (i == lineLen) break;
        count:
        if (line[i] == target[j]) {
            i++; j++;
        }
        else {
            i = i - j + 1;
            j = 0;
        }
    }
    return j == targetLen ? ans + 1 : ans;
}
```

统计分析结果：去掉符号为了更好地操作：

```
void Analyse(string tmp,int line,int func) {
    /**/tmp.erase(remove_if(tmp.begin(), tmp.end(), static_cast<int(*)(int)>(&ispunct)), tmp.end());
    for (auto x : targetword) {
        string target = x;
        target.erase(remove_if(target.begin(), target.end(), static_cast<int(*)(int)>(&ispunct)), target.end());
        int ans = KMP(tmp, target);
        if (ans) {
            if (word_count.find(x) != word_count.end()) {
                word_count[x].first += ans;
                word_count[x].second += (to_string(line) + " ");
            }
            else {
                word_count[x].first = ans;
                word_count[x].second += (to_string(line) + " ");
            }
        }
        else {
            word_count[x].first += ans;
        }
        ///*if (word_line.find(x) != word_line.end())*/ word_line[x].push_back(line);
    }
}
```

4、 程序运行方式简要说明。

首先，初始化三个容器。然后进入循环，先输入地址打开文件计算行数，获取

keyword：

```
//输入地址打开文本

while (1) {
    targetword.clear();
    keyword.clear();
    word_count.clear();
    cout << "|-------------------------------------------------------------------------------------------|" << endl;
    cout << "|----------------------------------文本检测程序---------------------------------|" << endl;
    cout << "|-------------------------------------------------------22336216 陶宇卓------------|" << endl;
    string filename;
    cout << "|-----------------------------请输入地址：(exit退出)-----------------------------|" << endl << endl;
    cin >> filename;
    if (filename == "exit") return 0;
    const char* file = filename.c_str();
    ifstream ReadFile;
    ReadFile.open(file);
        while (ReadFile.fail()) {
            cout << "|-----------------------------------打开失败！-----------------------------------|" << endl << endl;
            cin.ignore(10000, '\n');
            cin >> filename;
            const char* file = filename.c_str();
            ReadFile.open(file);

        }
        file = filename.c_str();
        //计算行数
        int countlines = CountLines(file);
        cout << "|-----------------------------------读取文本：-----------------------------------|" << endl << endl;
        for (int i = 1; i <= countlines; i++) {
            string oneline = Readline(file, i);
            cout << setw(2) << left << i << ":" << oneline << endl;
            GetKeyWord(oneline);
        }
```

根据输入的 1 或者 2 决定是否额外输入 targetword：

```
        //输入目标单词
        sort(keyword.begin(), keyword.end());
        cout << endl;
        cout << "|-------------------------1：查找所有单词 2：查找指定单词-----------------------|" << endl << endl;
        int choose = 0;
        cin >> choose;
        while (choose < 1 || choose > 2 || cin.fail()) {
            cin.clear();
            cin.ignore(10000, '\n');
            cout << endl << "|-----------------------------------输入错误！-----------------------------------|" << endl << endl;
            cin >> choose;
        }
        if (choose == 1) {
            cout << endl;
            cout << "|-----------------------------------字符集：-----------------------------------|" << endl << endl;
            for (auto x : keyword) {
                cout << x << " ";
            }
            targetword.assign(keyword.begin(), keyword.end());
        }
        else if (choose == 2) {
            cout << endl;
            cout << "|-----------------------------输入单词（以exit结束）：-----------------------------|" << endl << endl;
            string tar;
            while (1) {
                cin >> tar;
                if (tar == "exit") break;
                targetword.push_back(tar);
            }
            sort(targetword.begin(), targetword.end());
        }
```

最后调用 analysis 函数分析统计，得出结果后将 map 转化为 vector，用自定义的 Compare 类里的操作符重载函数对其排序并输出结果：

```
        //分析。。。
        for (int i = 1; i <= countlines; i++) {
            string oneline = Readline(file, i);
            if (choose == 1) Analyse(oneline, i, 1);
            if (choose == 2) Analyse(oneline, i, 2);
        }
        vector<pair<string, pair<int, string>>> word_counttmp(word_count.begin(), word_count.end());
        sort(word_counttmp.begin(), word_counttmp.end(), Compare());
        //result
        cout << endl;
        cout << endl << "|-----------------------------------检测结果：-----------------------------------|" << endl << endl;
        for (auto x : word_counttmp) {
            cout << x.first << " 出现了" << x.second.first << "次 , 在第 " << x.second.second << "行" << endl;
            cout << "|-------------------------------------" << endl;
        }
        word_counttmp.clear();
        system("pause");
        system("cls");
    }

    return 0;
}
```