《软件需求工程》实验三 实验报告

一、小组成员信息：

161220001 艾山江·吐尔逊

161220138 吴晗

161220179 周科

171860522 沈天琪

二、成绩分配比例：

每人各 25%

三、实验简介：

选定一个开源 IDE 项目，确定一个软件需求 R，从该软件需求提出开始，对软件需求进行跟踪分析。

四、实验过程：

本次实验选定 VS code 为目标 IDE 项目，确定此项目的一个软件需求为“smart case”。我们将从 VScode 的 GitHub 网址上获取关于此需求被提出时的讨论文本，接着识别出实现 smart case 的代码。后续的话在网站上可以得到开发者对于这个需求的更新、补丁等行为，从此类支持中获取需求变更的情况，选出一个较为具有代表性的需求变更，对其进行分析，识别出与需求变更有关的代码。最后总结出此需求的生命周期时间线。

五、实验任务分配：（具体完成内容见各自文件夹目录下的实验报告）

艾山江·吐尔逊：整理关于此需求被使用者提出以及其可行性被开发者讨论的相关细节。

吴晗：整理关于此需求如何被实现的细节以及找到其具体实现代码。

周科：整理给出需求 R 的全生命周期的时间线；总实验报告的整合与撰写。

沈天琪：整体实验思路构思；整理需求变更后的相关细节，并找到相应的实现代码。

六、实验结果展示：

0.需求“smart case”介绍：

Smart case”：如果查询包含大写字符，它将处理大小写敏感的查询;如果不包含大小写敏感的查询，它将处理大小写不敏感的查询。

例如，一个文档包含 asdf、asdF

启用“smart case”后，查询 asdf 匹配第 1 行和第 2 行。查询 asdF 只匹配第 2 行。

它也适用于正则表达式。regex 查询 asd\S 匹配第 1 行和第 2 行。查询 asd[A-Z]只匹配第 2 行。

引入一个设置：search.smartCase 或 editor.find.smartCase。这很容易与 ripgrep 挂钩，但是还需要在编辑器提供结果和未使用 ripgrep 时实现它。这对于非 regex 查询很容易。对于 regex 查询，ripgrep 之前尝试判断 regex 是否匹配任何大写字符。但是这产生了意外的结果，比如\w 触发大小写敏感，所以现在它只检查 regex 文本是否有任何未转义的大写字符(BurntSushi/ripgrep#717)。幸运的是，这很容易匹配。

另外，“区分大小写”的切换按钮可以是一个 3 模式开关，也可以启用智能大小写。

1.明确提出需求 smart case 的文本：

“我认为在开始时不区分大小写的匹配应该比在中间的大小写匹配更高。”

“有不同的上下文具有不同的理想排序/评分（文件 vs 命令 vs 方法），除此之外，每个人都有自己的自定义意见，什么应该放在第一位。”

2.获取需求的有关讨论文本：

a. 这感觉很奇怪。如果一个人从输入“line”开始，他会期望方法以 line 而不是 spline 开始。

还有一个完全匹配（不区分大小写）：

b. 我有很多问题需要改进，以实现快速开放，IntelliSense 也很相似，但目前使用自己的自定义解决方案。也许是时候转换成一个解决方案了。。。不幸的是，有不同的上下文具有不同的理想排序/评分（文件 vs 命令 vs 方法），除此之外，每个人都有自己的自定义意见，什么应该放在第一位。

c. 我对智能感知周围的所有问题都没有足够的了解，所以我对声音模型没有多大帮助，但是我认为在开始时不区分大小写的匹配应该比在中间的大小写匹配更高。

另外，Vim 的 smartcase（http://Vim.wikia.com/wiki/search）非常有用。如果完成输入有任何大 写字符，VSCode 也可以敏感地对大小写进行评分；如果输入没有大写字符，则可以不敏感地对大小写进行评分？

d. 我不知道这怎么会是 22153 的骗局。另一个问题讨论模糊匹配，其中这个问题讨论精确匹配大小写不敏感和不区分大小写的起始字符匹配。

对于.line->.line 给出的不区分大小写的示例，我希望其行为与不考虑大小写时的完全单词匹配相同。

如上所述，它比那更复杂，但同时这是我感觉完全直观的，所以我希望能做些什么。

显然，人们已经习惯了 vscode 现在如何对匹配项进行评分，因此，假设这是实现的，那么最好将其作为一个选项。智商。匹配词：真的或是什么的。

e. 我不认为 22153 涵盖了这个。当我打出一个词，如果有一个建议，这是一个确切的匹配，它应该有最高的分数。

我们可以重新打开这个，或者你有兴趣打开一个大问题，涵盖所有评分问题 @bpasero 建议？

我有很多问题需要改进，以实现快速开放，IntelliSense 也很相似，但目前使用自己的自定义解决方案。也许是时候转换成一个解决方案了。。。

3.给出实现需求的代码：

实现 search.smartCase 设置

https://github.com/microsoft/vscode/commit/6c6364d3b9340143da9de2101e702707b628b213

src/vs/platform/search/common/search.ts

+Line98: isSmartCase?: boolean;

+Line163: smartCase: boolean;

src/vs/workbench/parts/search/browser/searchViewlet.ts:

-Line33: import { ISearchProgressItem, ISearchComplete, ISearchQuery, IQueryOptions, ISearchConfiguration } from 'vs/platform/search/common/search';

+Line33: import { ISearchProgressItem, ISearchComplete, ISearchQuery, IQueryOptions, ISearchConfiguration, IPatternInfo } from 'vs/platform/search/common/search';

-Line983: const content = {

+Line983:const content: IPatternInfo = {
-Line988:wordSeparators:this.configurationService.getValue<ISearchConfiguration>
()editor.wordSeparators
+Line988:wordSeparators:
this.configurationService.getValue<ISearchConfiguration>().editor.wordSeparators,
+Line989:isSmartCase:
this.configurationService.getValue<ISearchConfiguration>().search.smartCase};

src/vs/workbench/parts/search/common/queryBuilder.ts
+Line73: this.resolveSmartCaseToCaseSensitive(contentPattern);
+Line100:

```
/**
	 * Fix the isCaseSensitive flag based on the query and the isSmartCase flag, for search providers that don't support smart case natively.
	 */
	private resolveSmartCaseToCaseSensitive(contentPattern: IPatternInfo): void {
		if (contentPattern.isSmartCase) {
			if (contentPattern.isRegExp) {
				// Consider it case sensitive if it contains an unescaped capital letter
				if (contentPattern.pattern.match(/([^\\]|^)[A-Z]/)) {
					contentPattern.isCaseSensitive = true;
				}
			} else if (contentPattern.pattern.match(/[A-Z]/)) {
				contentPattern.isCaseSensitive = true;
			}
		}
	}
```

src/vs/workbench/parts/search/electron-browser/search.contribution.ts:
+Line424: },

```
			'search.smartCase': {
				'type': 'boolean',
				'description': nls.localize('search.smartCase', "Searches case-insensitively if the pattern is all lowercase, otherwise, searches case-sensitively"),
				'default': false
```

src/vs/workbench/parts/search/test/common/queryBuilder.test.ts
+Line605:

```
	suite('smartCase', () => {
		test('no flags -> no change', () => {
			const query = queryBuilder.text(
				{
					pattern: 'a'
				},
```

```
            []);

        assert(!query.contentPattern.isCaseSensitive);
});

test('maintains isCaseSensitive when smartCase not set', () => {
        const query = queryBuilder.text(
            {
                pattern: 'a',
                isCaseSensitive: true
            },
            []);

        assert(query.contentPattern.isCaseSensitive);
});

test('maintains isCaseSensitive when smartCase set', () => {
        const query = queryBuilder.text(
            {
                pattern: 'a',
                isCaseSensitive: true,
                isSmartCase: true
            },
            []);

        assert(query.contentPattern.isCaseSensitive);
});

test('smartCase determines not case sensitive', () => {
        const query = queryBuilder.text(
            {
                pattern: 'abcd',
                isSmartCase: true
            },
            []);

        assert(!query.contentPattern.isCaseSensitive);
});

test('smartCase determines case sensitive', () => {
        const query = queryBuilder.text(
            {
                pattern: 'abCd',
                isSmartCase: true
```

```
                },
                []);

            assert(query.contentPattern.isCaseSensitive);
        });

        test('smartCase determines not case sensitive (regex)', () => {
            const query = queryBuilder.text(
                {
                    pattern: 'ab\\Sd',
                    isRegExp: true,
                    isSmartCase: true
                },
                []);

            assert(!query.contentPattern.isCaseSensitive);
        });

        test('smartCase determines case sensitive (regex)', () => {
            const query = queryBuilder.text(
                {
                    pattern: 'ab[A-Z]d',
                    isRegExp: true,
                    isSmartCase: true
                },
                []);

            assert(query.contentPattern.isCaseSensitive);
        });
    });

src/vs/workbench/services/search/node/ripgrepTextSearch.ts:
-Line426:       args.push(config.contentPattern.isCaseSensitive  ?  '--case-sensitive'  :  '--ignore-case');
+Line426: if (config.contentPattern.isSmartCase) {
        args.push('--smart-case');
    } else {
        args.push(config.contentPattern.isCaseSensitive  ?  '--case-sensitive'  :  '--ignore-case');
    }
```

4.明确需求变更的文本：

https://github.com/microsoft/vscode/issues?utf8=%E2%9C%93&q=is%3Aissue+smartCase+&tdsourcetag=s_pctim_aiomsg

smartCase
https://github.com/microsoft/vscode/issues/21362
20170227 功能请求
https://github.com/microsoft/vscode/issues/41119
为 smartcase 增加了配置项以控制这项功能是否启动
https://github.com/microsoft/vscode/issues/42684
smartcase 智能启动启用“区分大小写”时，应忽略 smartCase

https://github.com/microsoft/vscode/issues/42684
When Match Case is on, smartCase should be ignored
启用“区分大小写”时，应忽略 smartCase
需求变更提出 201802010656GMT+8

201802010753GMT+8
修复

201802020257GMT+8
修复验证通过

5. 给出实现需求变更的代码：

src/vs/workbench/services/search/node/ripgrepTextSearch.ts

```
function getRgArgs(config: IRawSearch) {
    const args = ['--hidden', '--heading', '--line-number', '--color', 'ansi', '--colors', 'path:none', '                --colors', 'line:none', '--colors', 'match:fg:red', '--colors', 'match:style:nobold'];
    /*
    if (config.contentPattern.isSmartCase) {
        args.push('--smart-case');
    } else {
        args.push(config.contentPattern.isCaseSensitive ? '--case-sensitive' : '--ignore-case');
    }
    */
    args.push(config.contentPattern.isCaseSensitive ? '--case-sensitive' : '--ignore-case');
```

6. 给出需求 R 的生命周期时间线。

2017.02.27 功能请求
~ 需求功能的讨论
2018.01.04 功能实现
~ 变更提出与讨论
2018.02.01 需求变更并实现（修复）
2018.02.02 修复验证通过