**《软件需求工程》实验一 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：**

本次实验我们实现的是对开源IDE项目“Visual Studio Code”进行软件需求的抽取与分类。我们运用实验材料中给出的思路一与思路三，分别对vscode进行了需求分析，并将两者的需求分析结果进行了相互印证与相互补充，综合整理出来的分析结果来作为我们此次实验的实验结果。

**四、实验过程：**

1.思路一部分：

利用爬虫程序爬取VSCode的GitHub项目中提取所有的issue信息，对爬取的文件进行信息提取，再利用词频统计程序对提取出来的有效信息进行词频分析，人工剔除无意义的高频词之后，基于词频结果分析软件需求。

2.思路三部分：

进入vscode的官方网站（https://code.visualstudio.com），利用官网主页的docs栏与 updates栏中提供的资源，通过功能倒推需求的方式，来进行软件的需求分析。

**五、实验任务分配：**

艾山江·吐尔逊：对提取出来的有效信息进行词频统计，人工剔除无效高频词，并进行需求分析与整理。

吴晗：阅读官网的updates部分，进行需求分析与整理。

周科：对爬取到的文件进行文档过滤与信息提取，提取有效信息交由后续处理；实验报告的整合与总结。

沈天琪：编写爬虫程序爬取VSCode的issue信息交由后续处理；阅读官网的docs部分，进行需求分析与整理。

**六、思路一代码实现：**

首先是从VSCode的GitHub项目中提取所有的issue。通常而言，GitHub中的issue是向开发者提出使用中所遇到的问题的途径，所以对此分析可以从中得到需求。由于GitHub向用户提供了官方API可以用于提取数据，所以可以依此编写批处理文件，利用curl软件获取数据。在这个任务中，代码编写并不是最大的难点，最大的难点是网络连接问题。最终解决方案是我将任务提交到了一个位于香港的服务器进行执行。

代码主要部分如下：

shell  
for (( i = 1; i <= 816; i++))  
do  
    echo "curl -4 -i -u [你的用户名]:[所对应的token] https://api.github.com/repos/microsoft/vscode/issues?state=all\&per\_page=100\&page=${i}\&direction=asc -o vscode${i}.txt"  
    curl -i -u [你的用户名]:[所对应的token] https://api.github.com/repos/microsoft/vscode/issues?state=all\&per\_page=100\&page=${i}\&direction=asc -o vscode${i}.txt  
done

由此得到的数据存储在vscode1文件夹目录下，然后需要对于从vscode网站中爬取到的816个issue文件进行文档过滤。可以看到的是，文档中各类url以及id信息属于上传者的个人信息部分，对本地实验是无用信息。本次实验的数据支持主要是由issue的“title”栏和“body”栏的具体内容构成。

由于文档中的title和body的信息存储格式均为"title": "…"和"body": "…"类型，故而选择正则表达式匹配来处理数据。构造出的正则表达式为“title/body”:\s”.\*”,利用regex\_search函数逐个处理这816个文档，将匹配到的title和body的具体内容输出到title.txt和body.txt这两个文件中。

主要部分代码如下：

smatch mat1;

regex pattern1("\"title\":\\s\"(.\*)\"");

string::const\_iterator start1 = allTitle.begin();

string::const\_iterator end1 = allTitle.end();

while (regex\_search(start1, end1, mat1, pattern1))

{

//cout << "1" << endl;

string msg(mat1[1].first, mat1[1].second);

fout1 << msg << endl;

start1 = mat1[0].second;

}

提取出有效信息以后，得到待分析的原始文件body.txt和title.txt。这里面隐含着开发者对该IDE的一些需求，我们希望从中获取到开发者需求中比较典型，搜索热度高的内容。

由于title.txt和body.txt文件里面中英文混用，并且还有许多无关的符号，网址，甚至别的国家的语言； 因此，先进行了语言分离，分理出三个文件，进行中英文的区分。

接着对上述三文件进行单词提取，并统计单词出现频率，画出频率分布直方图。

然后对数据进行分析，因为原数据并不为一个整体，因此没有选择主旨分析，权重分析，位置分析，而是选择了词频分析。根据词频分析结果进行简单的人工筛选，整理出一些热门需求关键词，部分列举如下：

中文需求部分：

文件，代码，提示，打开，自动，显示，输入，问题，出现，插件，更新，设置，中文，搜索，错误，终端，请求，版本，快捷键，扩展…

英文title需求部分：

backlog,file,vs，vscode,window,terminal,setting,work,june,march,support,extension,add,search,update,error,bug,command,tab…

英文body需求部分：

enabled,br,version,os,nsteps,reproduce,github,images,summary,app,insider,studio,issure…

通过词频分析结果获取到的可能需求如下：

代码提示功能

为什么不提供中文版，执行显示乱码

正则替换

帮助切换开发人员工具

安装插件时在搜索或下载插件页面好慢切换到另外窗口就马上把安装插件

搜索希望得到优化

折叠节点显示错误

功能建议文档修改状态的保存

格式化代码之后后面部分代码无法按照主题颜色显示全部显示为白色

共享进程意外终止请重新加载窗口以恢复

报告伪木马

关于“自动”：

自动增加括号

自动排版

保存时格式自动整理后和规则冲突的问题求助

多行注释自动添加\*号

开启了自动更新为什么更新后我以前的配置全部给重置了关键的问题

不能自动判断编码是或

自动补全智能提示

根据环境变量的配置自动找到解释器

粘贴代码自动缩进

代码智能提示自动导入功能

自动算法实现

....

关于“智能”：

我需要添加智能感知提示目前在扩展中的插件并不好用

关于“需求、要求”：

需求快速复制选中的内容

所有插件经常突然失效需要重启软件才行

我很需要能够支持java

需要重新整理下国际化语言包

为什么目录树里面的文件不能按住进行多选这个功能很需要啊

这个常驻左侧很麻烦也不美观还需要隐藏掉对我们这些两列编辑的简直太难受了

High DPI scaling for UI or settings for UI addition

Hide File Preferences Workspace Settings when not applicable

关于"smart":

schema less JSON intellisense could be smarter

Salsa unreachable code detected is not so smart

smartSelect is not working in html file?

Debugging with smartstep is super slow

smart code stepping

When using multiple carets, smart additions get in the way

Do something smarter when the integrated terminal is exited

CMD+D should be smarter

Smarter way to add (and autoclose) brackets

Smart Code Snippets (Complete the feature please)

Smarter search batching

Add SmartIndent capability

About smart tips and completion

Automatic Smart code folding (Freeze parent syntax node headers)

Smarter workspaceContains handling?

Smart loading of extensions based on manifest Backlog

try to be smarter about encoding detection

Smartscreen blocked download of VSCode installer from Edge

Suggestion About SmartExpandSelection

Should the output channel picker group logs smarter?

Smarter smart case search UI

**七、实验结果展示：(分类与整合后)**

**- 目标需求**  
    -Visual Studio Code是一个轻量级但功能强大的源代码编辑器，可在您的桌面上运行，并且可用于Windows，macOS和Linux。它具有对JavaScript，TypeScript和Node.js的内置支持，并具有丰富的其他语言（例如C ++，C＃，Java，Python，PHP，Go）和运行时（例如.NET和Unity）扩展的生态系统。  
  
**- 业务需求**  
    -智能代码补全  
    -简化的调试  
    -快速，强大的编辑  
    -代码导航和重构  
    -产品内源代码控制  
    -支持开源拓展插件  
    -键盘快捷键  
    -用于Windows，macOS和Linux

-显示在小地图的文本选择

在小地图概述看到选择区域。

-切换区域折叠键盘快捷键

切换折叠的快速扩展和折叠区域。

-源代码控制树视图

显示列表或新树视图中挂起的更改。

-在自定义工作目录中打开终端

为特定文件夹添加键盘快捷键。

-调整光标周围的行

保持你的光标在编辑器的中心。

-隐藏/显示状态栏项

只显示用户喜欢的状态栏项。

-在探索者中的缩进指南

清楚地突出项目的文件夹结构。

-更好的终端外壳选择器

容易地选择默认外壳在集成终端中使用。

-问题面板悬停命令

应用快速修复或打开一个窗口直接从悬停。

-树状用户界面的改进

改进了浏览器导航和过滤，水平滚动。

**- 功能需求**  
    -IntelliSense  
    -终端工具  
    -整理，多光标编辑，参数提示  
    -使用peek快速浏览源代码，并导航到definition  
    -编辑器内部的SCM支持，包括丰富的Git集成  
    -键盘快捷方式  
    -更改VS Code的键盘快捷键以匹配另一个编辑器  
    -修改默认键盘快捷键  
    -自动更新  
    -收集使用情况数据和崩溃报告信息  
    -VS Code 扩展，扩展集成到VS Code的UI，命令和任务运行系统中，通过VS Code的共享界面可以轻松使用不同的技术  
    -Windows作为Linux开发人员机器  
    -自定义标题栏  
    -触控栏支持（macOS）  
    -使用WSL  
    -自带入门手册  
    -根据当前上下文访问所有可用命令  
    -命令行界面  
    -更改语言模式  
    -改变你的主题  
    -调整设置  
    -添加JSON验证  
    -Zen模式  
        Zen Mode通过隐藏除编辑器之外的所有UI（没有活动栏，状态栏，侧边栏和面板），全屏显示并居中放置编辑器布局，使您可以专注于代码。  
    -自动保存  
    -集成终端  
    -并排编辑  
    -移至资源管理器窗口  
    -创建或打开文件  
    -导航历史  
    -为未正确检测到的文件创建语言关联  
    -防止覆盖在编辑器之外进行的更改  
    -列（框）选择  
    -多光标选择  
    -快速滚动  
    -上下复制行  
    -上下移动线  
    -缩小/扩展选择  
    -转到文件中的符号  
    -转到工作区中的符号  
    -导航到特定行  
    -修剪尾随空格  
    -代码格式化  
        格式化整个活动文件、格式化所选文本  
    -代码折叠  
    -导航到文件的开头和结尾  
    -并排Markdown编辑和预览  
    -速查参考  
    -搜索和修改  
    -任务执行器  
    -便携式模式  
        VS Code具有可移植模式，可让您将设置和数据保存在与安装相同的位置，例如在USB驱动器上。  
    -小地图  
        Minimap（代码大纲）为您提供了源代码的高级概述，这对于快速导航和理解代码很有用。文件的小地图显示在编辑器的右侧。您可以单击或拖动阴影区域以快速跳至文件的不同部分。  
    -缩进指南  
        VS Code可让您控制文本的缩进以及是否要使用空格或制表位。  
        VS Code分析您打开的文件并确定文档中使用的缩进。自动检测到的缩进将覆盖您的默认缩进设置。  
    -面包屑  
        编辑器在其内容上方有一个导航栏，称为“ 面包屑”。它显示当前位置，并允许您在文件夹，文件和符号之间快速导航。  
        面包屑始终显示文件路径，如果当前文件类型支持符号的语言，则符号路径直至光标位置。您可以使用“ 视图” >“ 显示面包屑”切换命令禁用面包屑。  
    -隐藏菜单栏（Windows，Linux）  
    -预览模式  
    -网格编辑器布局  
        默认情况下，编辑器组以垂直列布局（例如，当您拆分一个编辑器以将其打开到侧面时）。您可以轻松按自己喜欢的任何布局在垂直和水平方向上排列编辑器组  
    -热出口  
        默认情况下，VS Code会记住未保存的文件更改  
    -查找和替换  
        除了查找和替换为纯文本之外，“查找小部件”还具有三个高级搜索选项：相符、匹配整个单词、正则表达式  
    -多重文件编码支持  
    -快速文件导航  
    -多根工作区  
        您可以在具有多根工作区的Visual Studio Code中使用多个项目文件夹  
    -代码片段模板  
    -Emmet支持  
    -辅助功能  
        缩放、高对比度主题、键盘导航、标签导航  
 -HTML ARIA属性引用链接——从智能感知直接链接到ARIA文档。

-CSS属性补全包括分号——输入CSS属性时添加分号。

-CSS颜色变量预览-颜色变量完成显示颜色样本。

-改进的列断点UI -直接在源代码中查看可能的内联断点。

-调用堆栈视图中的内联调试操作——在悬停状态下保持调试操作在上下文中。

-远程资源管理器更新-资源管理器现在显示WSL发行版和存储库容器。

-保留全局搜索和替换的大小写——保留跨多文件搜索/替换的字母大小写。

-设置编辑器字符串数组验证——检查最小值、最大值、枚举值和glob模式。

-复制和恢复在diff编辑器-轻松复制或恢复删除的内容。

-转到行支持负行号-快速跳转到一个文件的结尾。

-MDN参考链接的HTML和CSS -链接到MDN文档直接从智能感知。

-添加丢失的等待快速修复-找到被忽略的异步代码等待。

-调试数据断点——调试器在跟踪值更改时中断。

-在设置界面中编辑字符串数组——直接从设置编辑器添加到字符串数组。

-在文件资源管理器中显示搜索结果-快速跳转到文件资源管理器中的文件位置。

-显示选择中的空白—仅针对选定的文本查看空白字符。

-查找并替换保护框—替换文本而不丢失字母框。

-终端搜索UX改进-更直观的自底向上的搜索顺序的终端。

-更好的SCSS @import导航- VS代码现在为SCSS文件检查更多的位置。

-顺序任务执行——控制任务和子任务执行的顺序。

-跳转到游标调试—在跳转到新位置时跳过代码执行。

-禁用调试控制台自动换行——允许将调试输出保持在一行。

-JavaScript/TypeScript夜间构建-新的扩展集成JS/TS夜间构建。

-新的Java安装程序-安装VS代码、Java扩展包和所需的Java依赖项。

-远程开发(预览)改进-保存到本地文件系统，拖放文件到远程

-JavaScript/TypeScript的智能选择——基于语言语义展开文本选择。

-将部分类型提取为自己的类型别名。

-更好的合并冲突显示-查看合并冲突的变化周围的源代码。

-折叠列表和探险家的所有命令-许多小但方便的改进之一。

-远程开发(预览)可在稳定-现在可以稳定使用远程开发扩展。

-选择默认格式程序-按项目和语言设置默认文档格式程序。

-进入定义位置-打开Peek视图和/或导航到主位置。

-转换为析构参数-新的TS重构使用命名的参数对象。

-对子会话的调试支持——参见在主调试会话下显示的子进程。

-启动带有调试会话的URI——在调试web服务器代码时自动打开浏览器。

-安装缺少的扩展依赖项-检测并提供安装缺少的依赖项。

-Node.js应用程序节点的远程调试。学习如何附加到Azure上运行的Node.js应用程序

-首选代码操作——首选(很可能)代码操作支持自动修复常见问题。

-调试控制台自定义——修改调试控制台的字体大小、字体族和行高。

-更好的智能感知ARIA属性-更详细的HTML可访问性属性描述。

-智能感知- Vetur扩展现在支持在Vue.js内插智能补全。

-新的“捆绑扩展”主题——使用webpack来提高扩展的加载时间。

-多行问题输出-在问题面板中可以看到完整的多行问题细节。

-查找所有引用历史-快速重新运行以前的引用搜索。

-HTML、CSS和JSON的语义选择——基于语言语义的扩展/收缩选择。

-集成终端回流支持-当面板宽度变化时，终端文本自动回流。

-基于命令的用户输入变量——用于任务和调试配置的自定义用户输入。

-更新的扩展API文档-完全重写与扩展指南。

-多行搜索改进——无需使用正则表达式即可轻松创建多行搜索模式。

-Linux上的自定义标题栏—Linux上的自定义标题和菜单栏现在是默认的。

-代码片段注释变量——代码片段变量为每种语言插入正确的注释。

-JS/TS回调显示-现在你可以看到匿名回调的上下文。

-JSDoc标记高亮显示——包括在JSDoc中标记代码块的语法高亮显示。

-简化的调试配置——更好的初始启动配置的默认设置和快速选择UI。

-在文件夹打开时运行任务——配置任务在首次打开项目文件夹时运行

-文件图标在智能感知-快速区分文件类型在智能感知。

-更好的macOS支持-全屏模式控制和Mojave暗模式支持。

-可折叠的堆栈框架-折叠无兴趣的堆栈框架，以清楚地看到您的代码。

-改进的加载脚本视图-显示相关资源，如原始源文件。

-更新的扩展样本-统一的风格和结构，自述，和API的使用。

-扩展的CI配方——Azure管道为扩展构建配方。

-CSS @import path补全-当你输入CSS, SCSS和更少的时候，导入路径建议。

-JSON条件求值——支持“if”、“then”和“else”模式关键字。

-内置的加载脚本视图-加载脚本资源管理器现在可用于调试扩展。

-快速修复问题面板-应用代码修复，同时审查警告和错误。

-Windows上的用户设置-在Windows设置期间不再需要仰角。

-终端列选择- Alt+单击可在集成终端中选择文本块。

-添加所有缺少的导入——在JavaScript/TypeScript文件中添加所有缺少的导入的操作。

-JSX标记完成—在JavaScript/TypeScript中使用JSX标记，就像在HTML中一样。

-子词导航-快速导航在camelCase的词。

-未使用的变量检测——未使用的变量在JavaScript/TypeScript文件中是灰色的。

-迁移到新的文件重构——很容易将类和函数拉到它们自己的文件中。

-在移动/重命名时更新导入—自动更新JavaScript/TypeScript中的导入路径。

-复制工作空间在新窗口-伟大的扩展超过多个mon设置。

-对预定义变量的智能感知——调试和任务配置文件中的变量建议。

-使用Docker和MongoDB -学习在VS代码中集成Docker和MongoDB  
   -代码导航（代码地图），代码名称、代码框架、模板自动提供补全，提醒  
   -支持开源拓展插件，支持远程和本地编译

   -键盘快捷键  
   -根据环境变量的配置自动找到解释器  
   -粘贴代码自动缩进，自动排版，代码智能提示自动导入功能  
   -自动算法实现，按权重列出代码错误的可能解决方案  
**- 性能需求**  
    -轻量级  
    -快速启动和运行  
 -系统资源占用率低，系统兼容性高（中端机器上能良好的运行），分布式开发  
**- 约束与限制**  
    -使用C++开发  
    -用于Windows，macOS和Linux  
    -各个平台都有对应安装文件  
    -在大多数可用的硬件和平台版本上运行  
    -Visual Studio Code下载文件<100 MB，磁盘占用空间为200 MB。  
    -最低硬件要求：  
        -1.6 GHz或更快的处理器  
        -1 GB的RAM  
    -需要.NET Framework 4.5.2（Windows）  
    -GLIBCXX版本3.4.15或更高版本 以及 GLIBC 2.15版或更高版本 （Linux）  
    -更新频率约每月一次  
    -Visual Studio Code建立在Electron之上，并受益于Chromium的所有网络堆栈功能

**-其他需求**

-VS代码图标仓库-官方产品图标可用于扩展的作者。

-Alpine Linux发行版支持——既支持远程WSL，也支持容器扩展。

-完整的产品图标刷新-新的现代图标跨越所有的VS代码。

-远程开发教程-在SSH和WSL上工作的新逐步教程。

-面包屑导航默认显示-通过面包屑快速进入文件和符号。

-集成终端支持真彩色-正确呈现真彩色(24位)转义序列。

-易于显示语言选择-快速选择之间的安装地区。

-可配置的智能感知显示——过滤掉关键字等元素或删除显示的图标。

-新的Python烧瓶教程-使用VS代码来创建和调试Python烧瓶web应用程序

-智能感知提示系统  
    -界面资料卡自动隐藏  
    -快速复制  
    -智能代码步进、智能缩进  
    -自动判断编码，算法，框架，语言正确性  
   -不同程序设计语言之间的转换  
   -代码风格养成、记录、自动匹配  
   -智能日志生成，管理  
   -在手机上可以实现代码查看、管理  
   -格式化代码之后后面部分代码无法按照主题颜色显示全部显示为白色  
   -共享进程意外终止请重新加载窗口以恢复  
   -报告伪木马  
   -智能化搜索，检测，智能推荐