**实验二：需求排序**

小组成员：171860566 王盼盼

171860565 刘振国

成绩比例1:1

**实验思路**

选定一个开源 IDE 项目，确定可能的信息来源，获取有效信息，对所获取的

需求进行优先级排序。

（1） 确定 IDE 项目；

（2） 明确信息源；

（3） 获取需求；

（4） 提出一种方法，对需求进行优先级排序。

（5） 对所提方法的效果进行分析。

**数据获取**

选择的IDE仍然是vscode

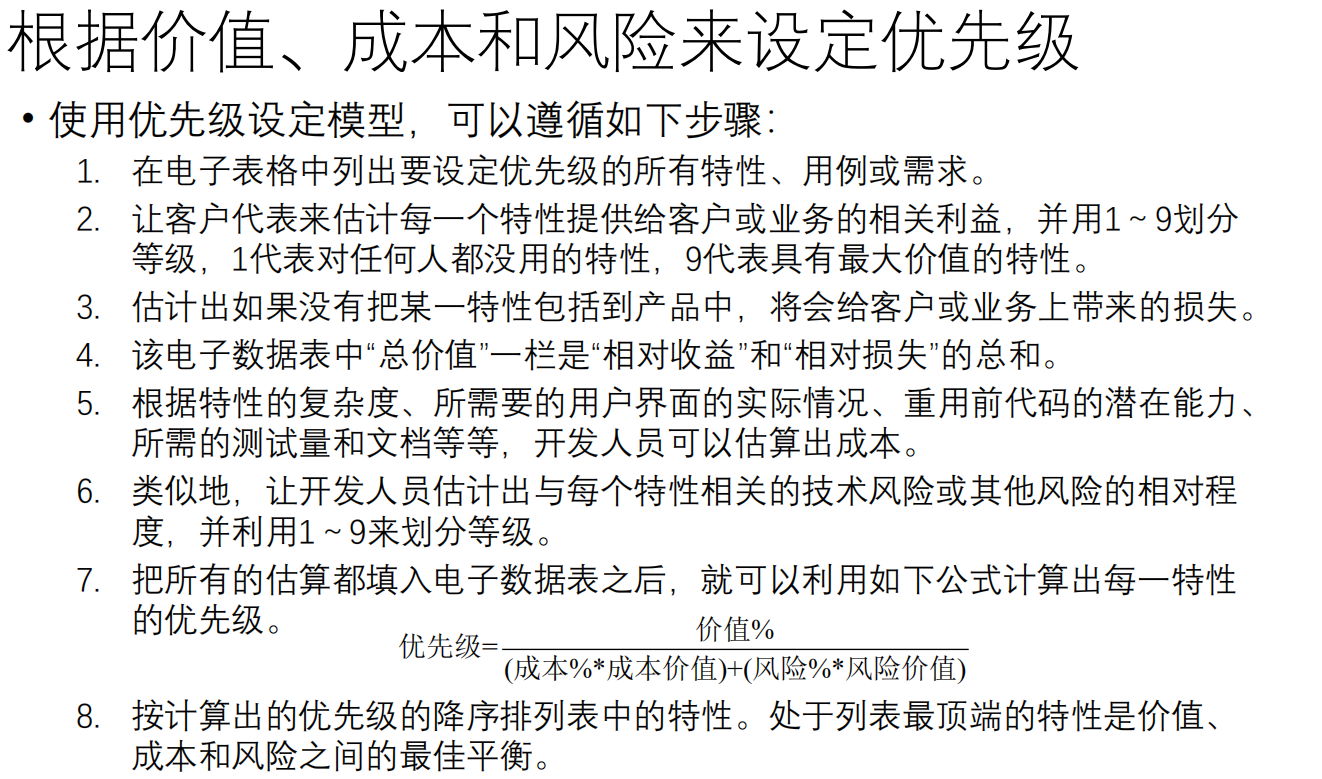
根据实验指导中思路一的提示，我们明确vscode托管在在GitHub上的地址作为信息来源，实验的思路是对该项目中的Pull Request（地址：https://github.com/microsoft/vscode/pulls）作为一种需求，获取相关信息。

我们发现这些PR的标题可以概括相应的需求信息，为了获得众多的标题信息，我们使用Python语言写代码（文件：vs\_pr\_get.py）来抓取这些信息并存入文件（pulls3.txt）。

之后利用实验一词频分析的思路，这里不做过多赘述

**需求排序**

优先级排序基于下面的算法：



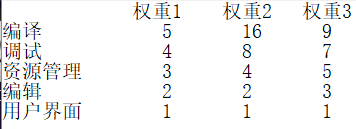
一名组员扮演客户代表，一名组员扮演开发人员代表，分别为每一个需求特性打分，利用上面的算法计算出每一个需求特性的基础优先级，然后乘以该需求所在大类的权重，得到最终的优先级。为了验证我们方法的有效性，我们设置了三种大类权重分配方式，计算出三种情况下的优先级，进行对比。排序结果在lab2\_sort.xlsx文件中。

**方法效果分析**

起初我们的想法是直接按照打分情况计算出基础优先级即可，然而结果却很不理想，因为有些需求很重要但它的成本相对较高，这样会使得它最终的优先级反而低于那些并不是很重要但是成本却很低的需求，这种情况显然很不合理。此外，对于优先级排序这种主观性很强的任务，如何评判某种方法的好坏标准不一，单独对一项结果进行排序分析难以做到，但是我们却可以对比不同结果的好坏，我们可以很容易找到合理与不合理的地方。

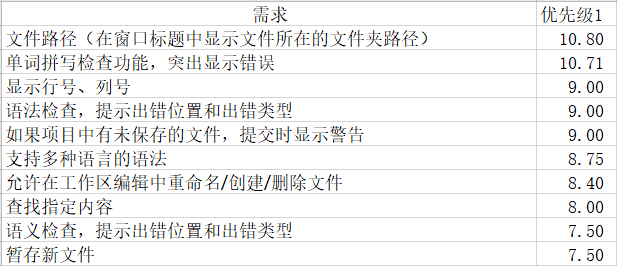
于是我们提出了先按照最初的想法计算出基础优先级，再用基础优先级乘以该需求所在大类的**权重**，得到最终的优先级。然后，在此基础上，我们横向对比三种权重的不同结果，从而得到某些结论。

基于对IDE的认知，我们认为在需求方面：**编译>调试>资源管理>编辑>用户界面**

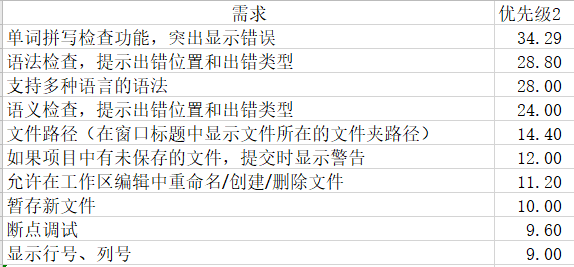


分别对三种优先级进行排序，提取出每种情况下的Top10，结果如下：

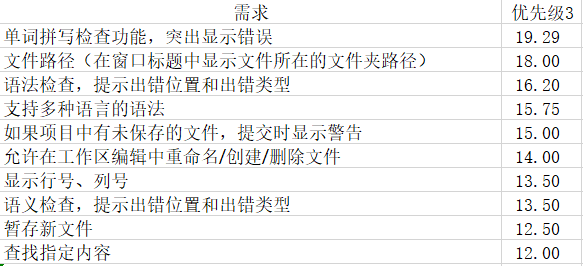
优先级1：



优先级2：



优先级3：



从上面的结果中可以看出，最高优先级是IDE所必备的功能，如：编译、汇编、文件、编辑等功能，这符合程序员对于一个IDE的客观要求。有些功能虽然使用起来非常的便捷，如查找、替换指定内容等，但没有它们并不会使IDE无法正常工作，因此它们的优先级处于中上的位置。综合来看，需求优先级的计算，必备需求的优先级权重较高，尽管它的成本和风险可能也较高，但这种成本和风险是开发人员必须克服的。

这些结果也会有具体排名上的不同，对于不同的权重分配问题，会引起不同的排名结果，但它们在一定方面都具有合理性，这取决于不同项目对实际需求的重视程度。

**仓库文件说明**

***data***

word\_frequency.txt 词频统计结果

pulls3.txt PR爬取结果

lab2\_sort.xslx 基于不同大类权重的成本风险优先级排序结果

***code***

Word\_frequency.cpp 词频统计程序

vs\_pr\_get.py vscode的PR标题爬取程序