**开源软件基础大作业**

**对 crystal 项目历史提交信息的分析统计**

组 长 姓 名： 李姝妮

学 号： 20222241375

组 员 姓 名： 王雨晴

学 号： 20222241402

组 员 姓 名： 尹奕昂

学 号： 20222241307

组 员 姓 名： 吴越秀

学 号： 20222241225

组 员 姓 名： 汪敬舒

学 号： 20222241291

完 成 日 期： 2025.1.24

大连理工大学

Dalian University of Technology

# 工作分配

李姝妮：组长，测试模块。

王雨晴：组员，数据爬取模块。

尹奕昂：组员，可视化模块。

汪敬舒：组员，数据清洗模块。

吴越秀：组员，文档撰写。

# 研究背景及意义

## 1.1 研究背景

Crystal 项目，是一个名为 Crystal 的编程语言相关的仓库，Crystal 仓库包含了 Crystal 编程语言的源代码、编译器、标准库等核心组件。其高性能编程，易学易用，开源社区支持的特点吸引了我们的注意。本项目旨在对 crystal 项目的历史提交信息进行深入分析统计，通过 Python 编程实现数据爬取、数据清洗、可视化等处理，并测试验证，挖掘该开源软件仓库的规律和特点。通过对数据的详细分析，我们可以更好地理解该项目的发展历程和演化现象，为相关研究和决策提供有力支持。

## 研究意义

对 crystal 项目历史提交信息的分析可以为项目的管理者提供决策依据，帮助开发者更好地了解项目的发展动态，同时也为开源软件研究领域提供了一个具体的案例分析。

## 1.3 挑战

1. 爬取数据信息不全，存在缺失值。

2. 数据清洗没有统一标准。

3. 可视化变量的选定。

4. 测试的准确性评估。

5. 文档的可读性。

# 2 需求分析

## 2.1 功能需求

1.数据爬取：爬取crystal项目的历史提交信息，包括提交内容，提交类型，浏览量和提交实际。

2.数据清洗：去除重复记录，处理乱码，过滤无效数据，转换日期格式等。

3.数据可视化：绘制各种图表，直观展示数据的分布和趋势，包括更新类型的分布、浏览量的分布、更新数量的时间趋势、高浏览量的更新内容、浏览量与更新时间和更新类型的关系等。

4.测试：确保数据爬取、清洗和可视化的代码能够正常运行，数据处理的结果符合预期，并且代码具有一定的鲁棒性，能够处理各种异常情况。

## 2.2 性能需求

数据处理和可视化过程应具有较高的效率，能够在合理的时间内完成。

## 2.3 可维护性需求

代码应具有良好的结构和注释，便于后续的维护和扩展。

# 3 实施方案

## 3.1 总体架构设计：

本项目旨在对 Crystal 编程语言的 GitHub 仓库进行全面的分析。主要分为数据爬取、数据清洗、数据可视化、测试、文档撰写五个模块。以确保能够高效地提取关键信息，并为 Crystal 项目的发展提供数据支持。以下是每个模块的详细设计。

## 3.2 数据爬取：

1.确定数据源：通过调研和分析，找到包含 crystal 项目历史提交信息的数据源，如项目的官方网站、代码托管平台的 API 等。

2.发送请求：使用requests库发送 HTTP 请求，获取网页的 HTML 内容或 API 返回的 JSON 数据。

3.解析数据：对于 HTML 内容，使用BeautifulSoup库进行解析，通过选择器或 XPath 定位到所需的数据元素；对于 JSON 数据，直接使用 Python 的内置json模块进行解析。

4.数据存储：将解析后的数据存储到本地文件（如 CSV 文件）中，以便后续的数据清洗和分析。

## 3.3 数据清洗：

1.去除重复行：使用pandas库的drop\_duplicates()方法去除数据中的重复行。

2.处理乱码：定义isLegal()函数，检查非 ASCII 字符比例、无效的 Unicode 字符组合以及孤立的控制字符，过滤掉可能的乱码数据。

3.过滤无效数据：检查View Count的值是否大于 0，过滤掉浏览量为 0 或负数的数据。

4.日期格式转换：将Update Time列的数据格式转换为日期格式，使用pandas库的to\_datetime()方法，并处理可能的日期转换错误。

## 3.4 数据可视化：

1.更新类型分布：使用 Seaborn 的 countplot 绘制更新类型的柱状图。

2.浏览量分布：使用 Seaborn 的 histplot 绘制浏览量的直方图。

3.更新数量时间趋势：按月份聚合数据，使用 Pandas 的 plot 方法绘制更新数量的时间趋势线图。

4.高浏览量的更新内容：筛选出浏览量前 10 的更新内容，使用 Seaborn 的 barplot 绘制条形图。

5.浏览量与更新时间的关系：使用 Seaborn 的 scatterplot 绘制散点图。

6.浏览量与更新类型联合分析：使用 Seaborn 的 boxplot 绘制箱线图。

## 3.5 测试：

测试目标：

确保数据爬取、清洗和可视化的代码能够正常运行，数据处理的结果符合预期，并且代码具有一定的鲁棒性，能够处理各种异常情况。

具体使用以下测试方法：

1.单元测试：使用unittest或pytest等测试框架对数据处理函数进行单元测试，确保每个函数的功能正确。例如，对isLegal()函数进行测试，验证其对不同类型文本的判断结果是否正确。

2.集成测试：对整个数据处理流程进行集成测试，包括数据爬取、清洗和可视化的各个步骤，确保整个流程能够顺利完成，并且输出的结果符合预期。

3.异常测试：针对可能出现的异常情况进行测试，如网络请求失败、数据格式错误等，确保代码能够正确处理这些异常，避免程序崩溃。

## 3.6 文档撰写：

1.文档目标：

为项目编写详细的文档，包括项目概述、数据爬取方法、数据清洗步骤、数据可视化结果以及代码说明等，方便其他开发者理解和使用本项目。

2.文档内容：

介绍项目的背景、目标、主要功能和使用方法，提供数据爬取、清洗和可视化的代码示例，以及相关的测试说明。

技术文档：编写技术文档，详细描述数据爬取、清洗和可视化的技术细节，包括使用的工具和技术、数据处理流程、可视化图表的含义等，为其他开发者提供技术参考。

3.文档格式：

Word

# 4 结果分析

## 4.1 更新类型分布

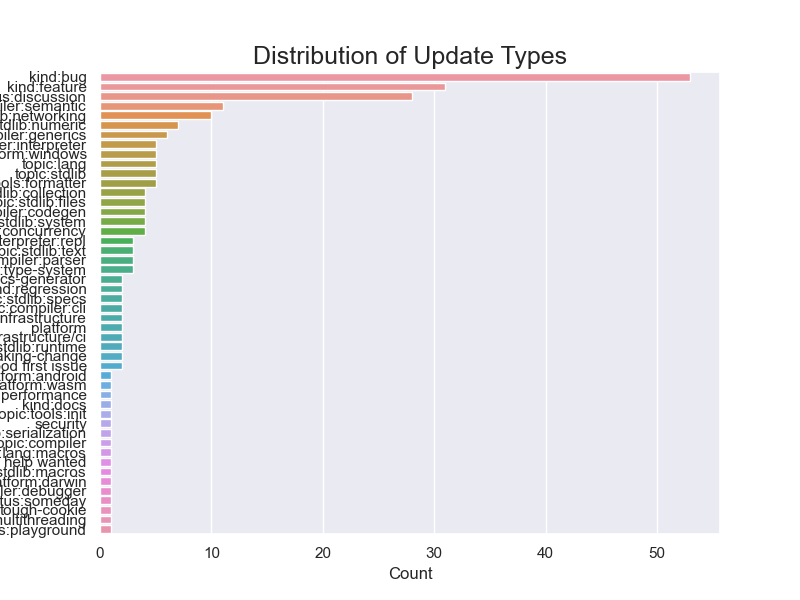


图4.1

通过更新类型的分布柱状图可以看出，不同更新类型的数量存在明显差异，某些更新类型出现的频率较高，而有些则相对较少。这可能与项目的开发重点和需求有关。

## 4.2 浏览量分布

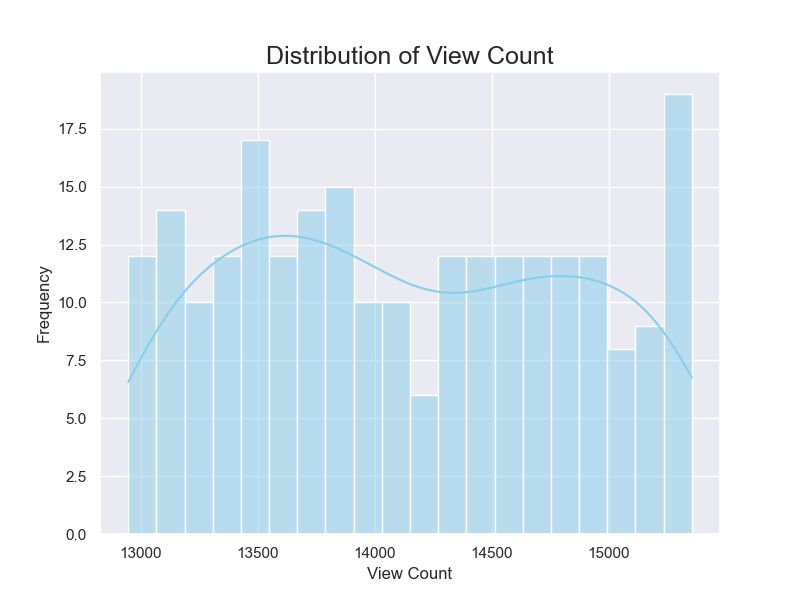


图4.2

浏览量的直方图显示，浏览量的分布呈现出一定的偏态，大部分更新的浏览量较低，只有少数更新获得了较高的浏览量。

## 4.3 更新数量时间趋势

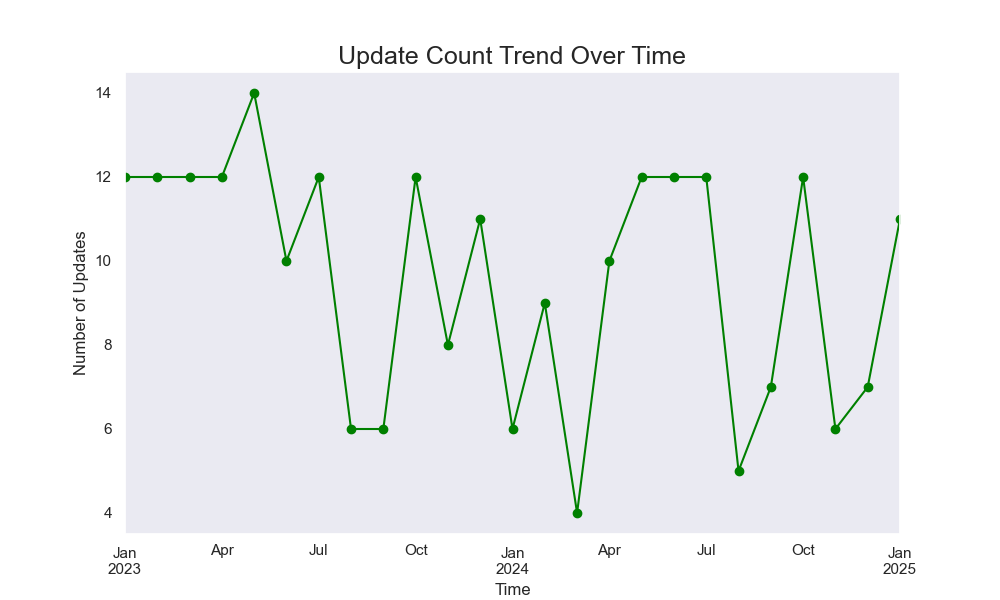


图4.3

更新数量的时间趋势线图反映了项目的活跃度随时间的变化情况。可以发现，在某些时间段内，项目的更新数量较多，而在其他时间段则相对较少，这可能与项目的开发阶段和团队的工作安排有关。

## 4.4 高浏览量的更新内容

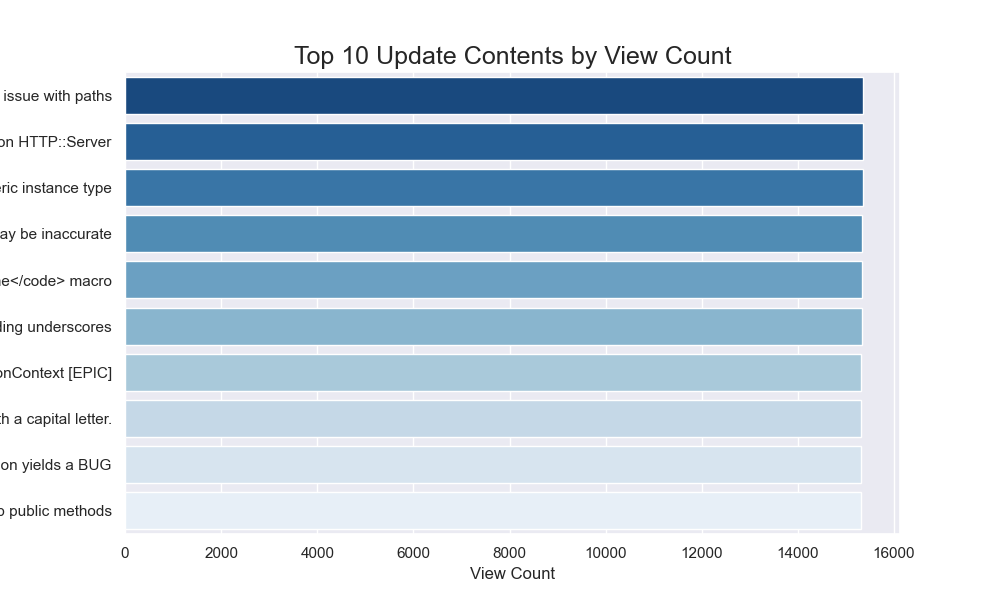


图4.4

高浏览量的更新内容条形图展示了浏览量排名前 10 的更新内容，这些内容可能是项目的重要更新或者受到了用户的广泛关注。

## 4.5 浏览量与更新时间、更新类型的关系

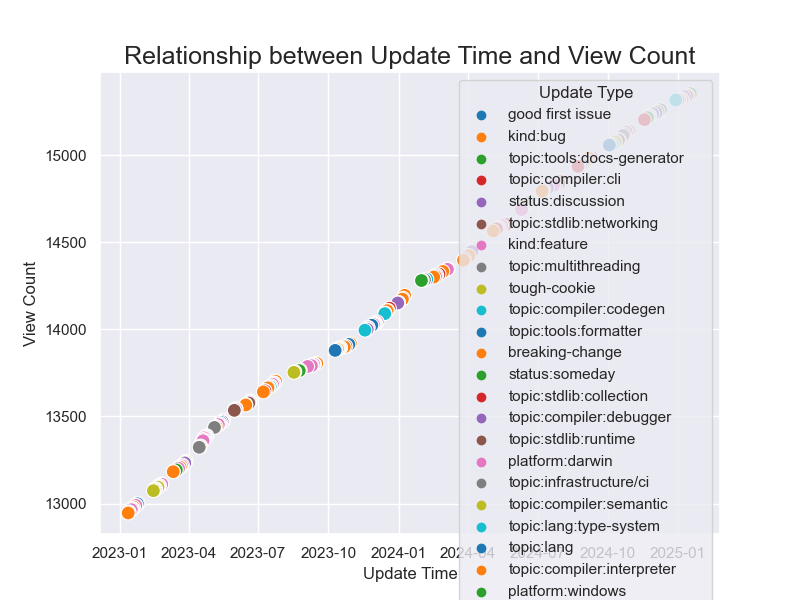


图4.5

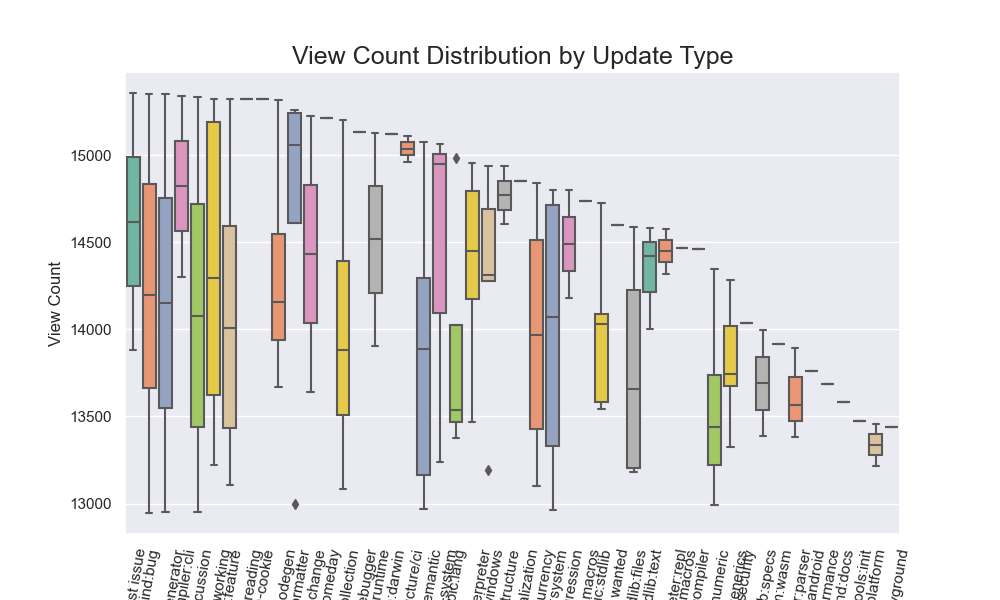


图4.6

散点图和箱线图分别展示了浏览量与更新时间、更新类型之间的关系。从散点图中可以看出，浏览量在不同的更新时间点上存在一定的波动；箱线图则显示了不同更新类型的浏览量分布情况，某些更新类型的浏览量中位数较高，而有些则较低。

# 5 结论

## 5.1 总结

通过对 crystal 项目历史提交信息的深入分析统计，我们利用 Python 语言在数据爬取、清洗、可视化及测试等环节进行了全面且系统的工作，成功揭示了该项目在多个维度上的规律与特点。

本研究不仅为 crystal 项目的管理和开发提供了有价值的参考，还为开源软件仓库的分析提供了一套切实可行的方法和典型案例。通过本项目的实践，我们展示了如何运用 Python 技术进行数据的收集、处理、可视化和分析，为其他开源项目的研究提供了借鉴。未来，我们可以进一步拓展分析维度，结合更多的数据指标，如代码复杂度、开发者贡献度等，对开源软件仓库进行更深入、全面的研究，为开源社区的发展提供更有力的支持。

## 5.2 未来方向

1.爬取更多的信息，挖掘其中蕴含的规律。

2.尝试对该项目进行模糊测试。

3.提升分析的代码性能。