**哈尔滨工业大学 计算学部**

**2024年秋季学期《开源软件开发实践》**

**Lab 2：开源软件开发协作流程**

|  |  |  |
| --- | --- | --- |
| **姓名** | **学号** | **联系方式** |
| 吴俣辰 | 2022211972 | 19955918221 |

**目 录**

[1 实验要求 1](#_Toc1931083871)

[2 实验内容1 发送pull request 1](#_Toc806262786)

[2.1 fork项目 1](#_Toc236831732)

[2.2 git操作命令 1](#_Toc1143721833)

[2.3 代码修改 1](#_Toc406722934)

[2.4 测试通过截图 1](#_Toc351903337)

[3 实验内容2 接受pull request 1](#_Toc269421121)

[4 实验内容3 github辅助工具 1](#_Toc1265252771)

[4.1 熟悉GoodFirstIssue工具 1](#_Toc720249603)

[4.2 安装并使用Hypercrx 2](#_Toc2017243129)

[4.3 利用OpenLeaderboard工具 2](#_Toc1480933914)

[5 小结 3](#_Toc720823868)

[文档全部完成之后，请更新上述区域]

# 实验要求

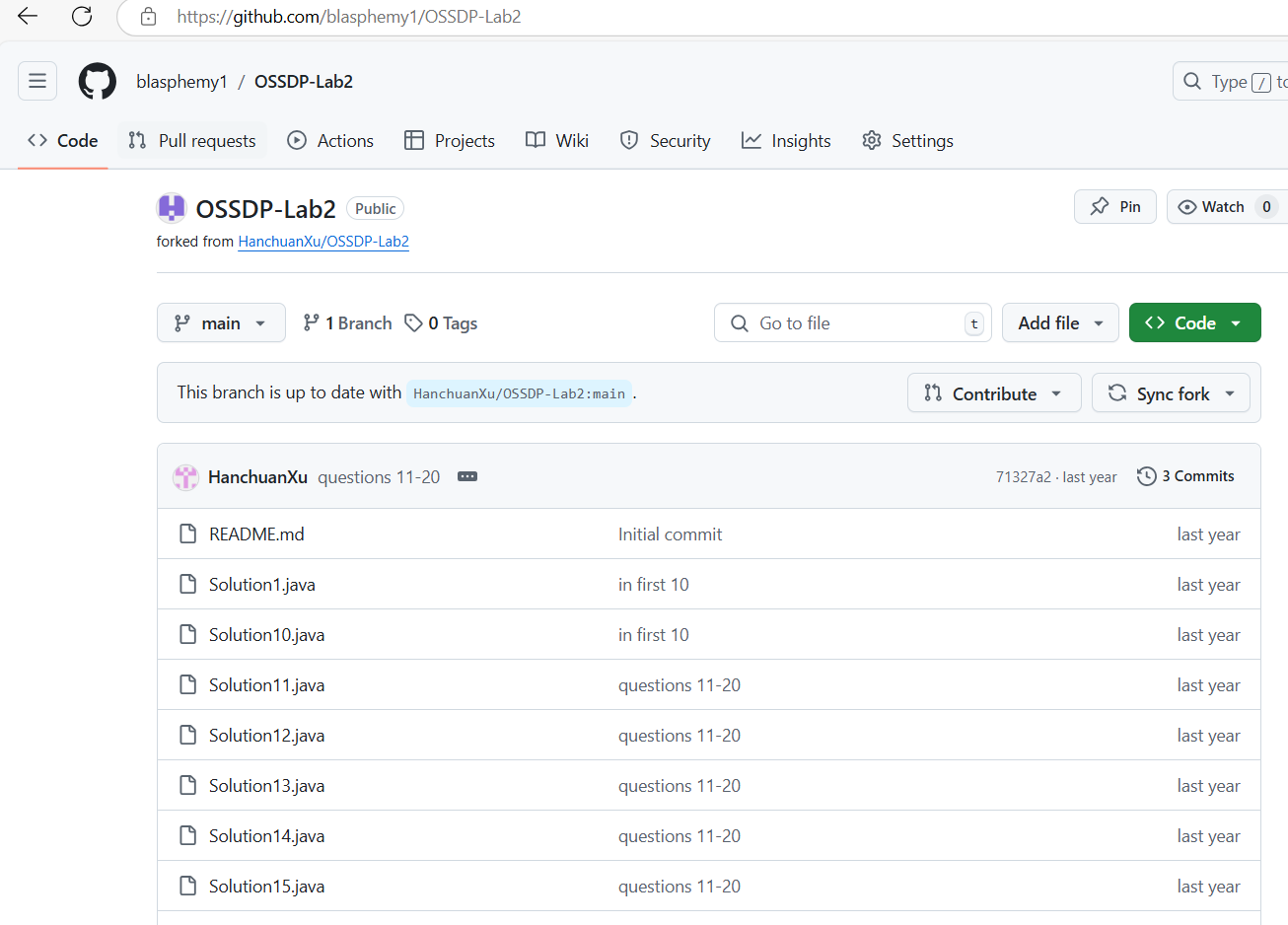
▪ 了解和掌握基于代码托管平台的开源软件协作开发过程

▪ 掌握基于github的软件项目协作开发命令和方法

▪ 熟悉几个github中常用开源软件开发工具

# 实验内容1 发送pull request

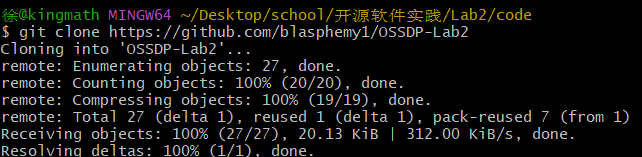
## fork项目



## git操作命令

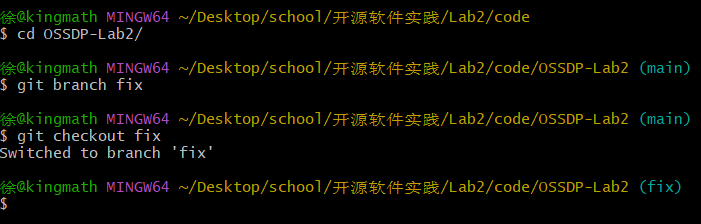
Clone github上fork后的仓库到本地

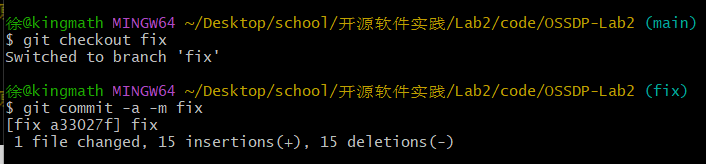
git clone [blasphemy1/OSSDP-Lab2 (github.com)](https://github.com/blasphemy1/OSSDP-Lab2)

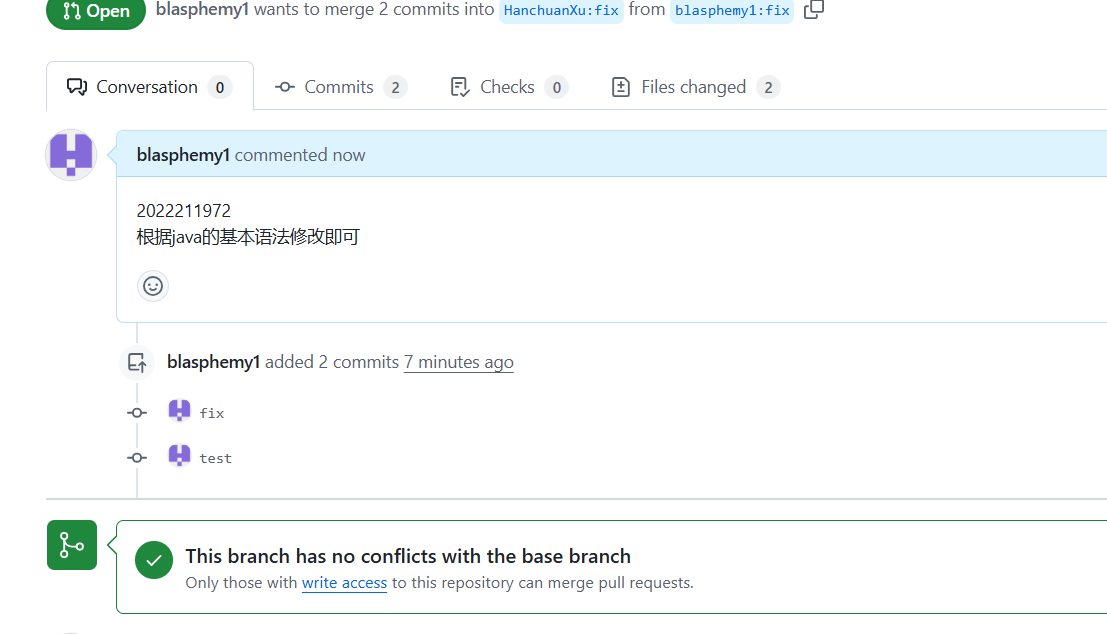


创建fix分支，在fix分支上修改所分配题目代码中的所有错误，编写测试类，

测试通过无误后，提交到本地仓库并推送到github上







## 代码修改

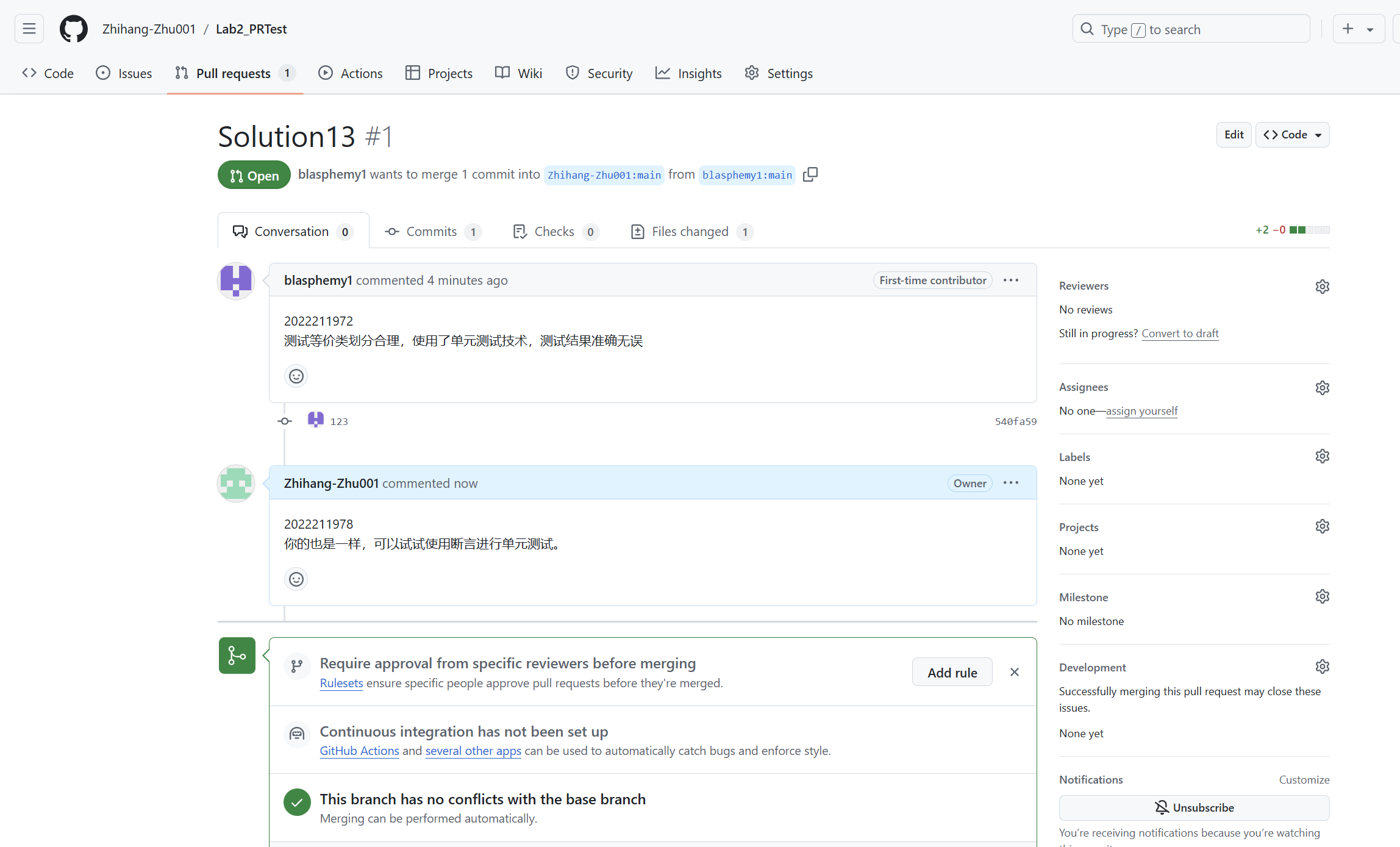
class Solution13{  
 public void sortColors(int[] nums) {  
 int n = nums.length;  
 int ptr = 0;  
 for (int i = 0; i < n; ++i) {  
 if (nums[i] == 0) {  
 int temp = nums[i];  
 nums[i] = nums[ptr];  
 nums[ptr] = temp;  
 ++ptr;  
 }  
 }  
 for (int i = ptr; i < n; ++i) {  
 if (nums[i] == 1) {  
 int temp = nums[i];  
 nums[i] = nums[ptr];  
 nums[ptr] = temp;  
 ++ptr;  
 }  
 }  
 }  
}

## 测试通过截图



# 实验内容

# 接受pull request



# 实验内容 github辅助工具

**4.1熟悉GoodFirstIssue工具**

**阐述如何使自己的开源项目被此网站收录：**

**1.至少有3个good first issue**

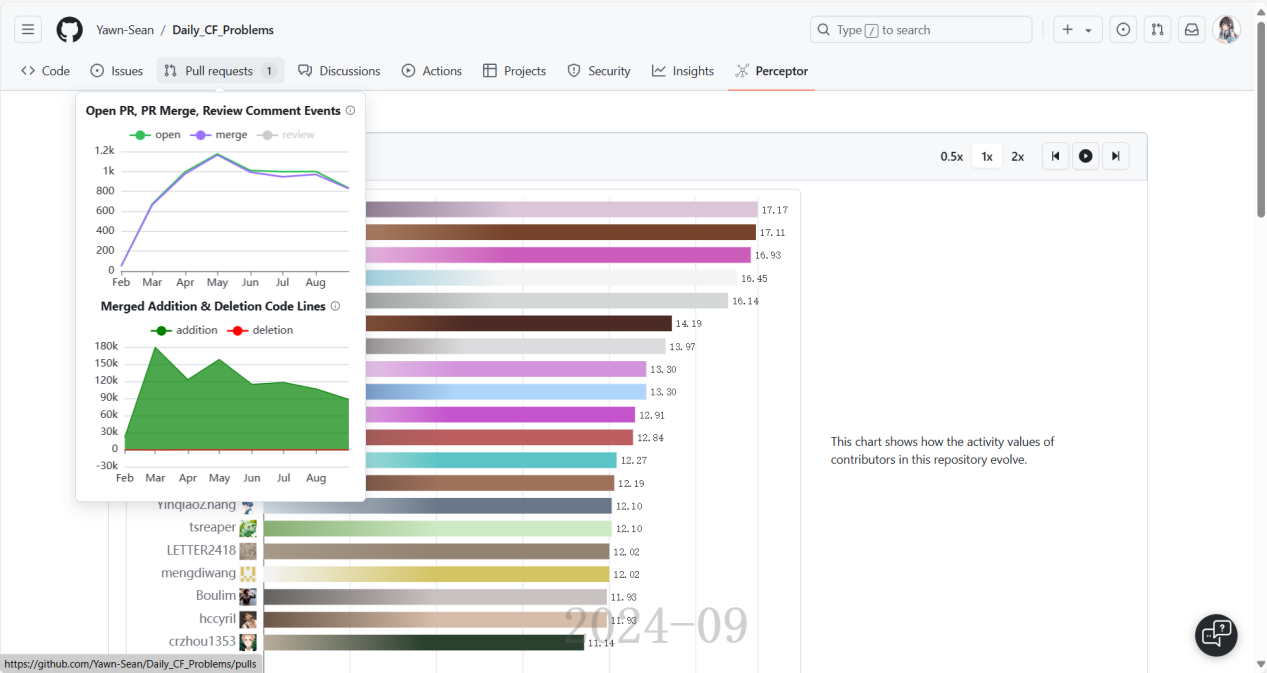
**2.至少有10名贡献者**

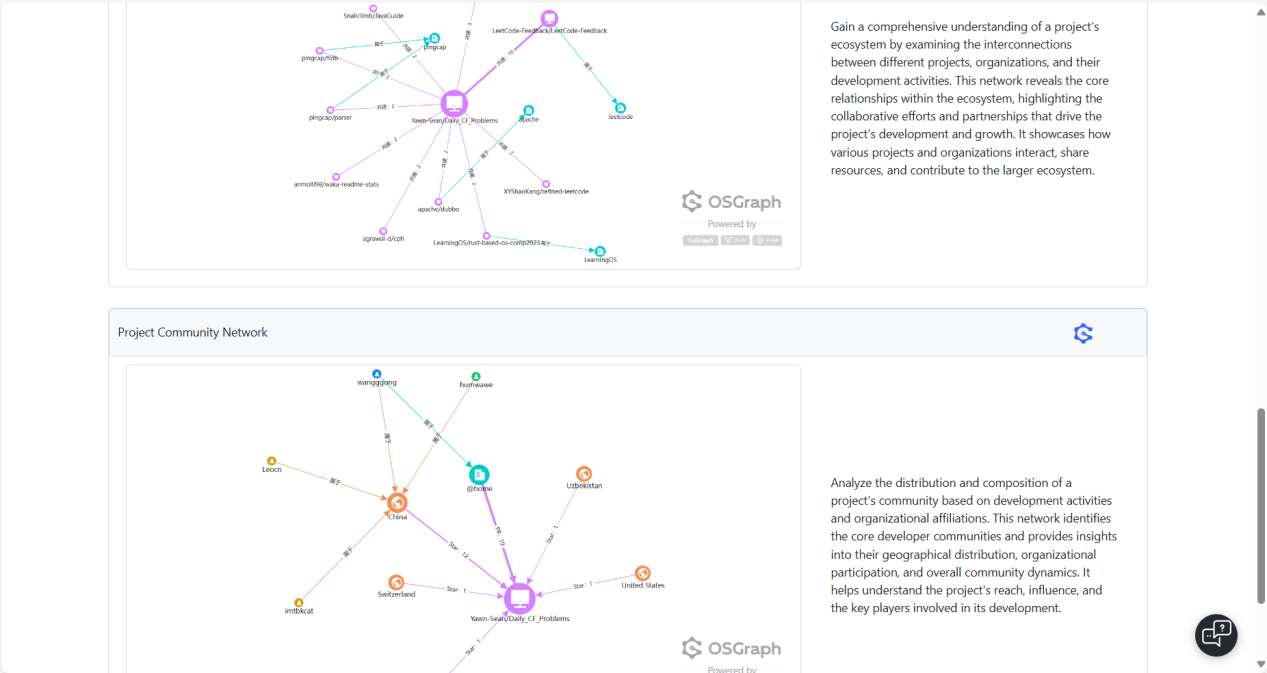
**3.有详细的readme.md文档介绍和新人友好的为新贡献者提供指南的 CONTRIBUTING.md 文件**

**4.2安装并使用Hypercrx**

**安装Hypercrx插件**

**找到项目<https://github.com/Yawn-Sean/Daily_CF_Problems.git>**





**4.3利用OpenLeaderboard工具**

**利用OpenLeaderboard工具了解github上开源项目的统计情况。**



**阅读实验手册中指定的三篇文章，概括开源项目活跃度、影响力指标、价值流网络的含义，OpenRank的计算原理。**

**开源项目活跃度：开发频率、问题管理、贡献者的参与、用户互动**

**影响力指标：星标数量、分叉数量、外部引用、下载次数**

**价值流网络：资源流动、参与者角色、互动关系、价值创造**

**OpenRank计算原理：指标选择（活跃度、影响力、社区参与）、各指标权重分配，标准化处理。**

# 小结

在当今软件开发的蓬勃发展中，开源软件不仅促进了技术的创新，也为开发者之间的协作提供了新的可能性。Git作为一种分布式版本控制工具，在开源软件开发中扮演着至关重要的角色。利用Git实现开源软件开发协作流程的过程中，我深刻体会到了团队合作、代码管理以及知识共享的重要性。

首先，Git为开发团队提供了高效的版本管理功能，使得多位开发者可以在同一项目中平行工作。通过分支管理机制，团队成员能够在不同的分支上进行功能开发或错误修复，避免了代码冲突的发生。这种并行协作的模式，极大地提高了开发效率，也使得项目能够在短时间内迭代更新。

其次，Git的协作流程强调了代码审查和持续集成的重要性。在开发过程中，通过提交请求（Pull Request）进行代码审查，不仅可以发现潜在的问题，还能促进团队内部的知识共享与技术交流。这一过程增强了团队成员之间的沟通，提升了代码质量，确保了软件的可靠性。

最后，使用Git进行开源软件开发深化了我对文档及注释重要性的理解。在协作中，清晰的提交信息和详尽的文档能够帮助其他开发者更好地理解项目的进展和设计思路。因此，良好的文档习惯成为提升团队协作效果的关键因素。

综上所述，利用Git实现开源软件开发协作流程，不仅提高了开发效率和代码质量，也促进了团队的紧密合作和知识的共享。这一过程的收获，不仅体现在技术能力的提升，更在于促进了合作精神与开源文化的深入理解。未来，我将继续在这一领域探索与实践，以期更好地参与开源社区的建设与发展。