OJ 大作业报告

林奕辰 2021010550 lin-yc21@mails.tsinghua.edu.cn

一、前言：

大作业 oj-lin-yc21-master 包含了基础功能（40）+ cli 前端（10）+ 多比赛支持（10）+ 持久化存储[json]（5）+ 非阻塞评测（10）+ 资源限制进阶（10）+ 打包测试（5）+ special_judge（10）+ 竞争得分（10），助教测试中，能支持大部分的基础测试，但是部分的 corner case 会挂。

二、程序结构：

①、lib.rs:存储全局变量，包括用户、问题、比赛的有关记录等；

②、api_analysis.rs:存储了包括 api 接口的所有所需的 json 相关结构体和相应的结构体函数，主要用于简化 main.rs，并且承担算法和计算的功能；

③、config_analysis.rs:存储了关于 config 设置相关的所有所需的 json 相关结构体和相应结构体函数；

④、trail_terminal.rs:存储了包含运行处理 post_jobs 等的相关算法函数，目的是为了优化 oj/main.rs 的框架并且减少 main 中的不必要的行数；

⑤、bin/oj/main.rs:OJ 的服务端，包含运行所需的 App，其 service 内部含有一系列相关函数，包括所需的 post、get、put 类型函数，各函数功能不同，风格相同。

⑥、bin/cli/main.rs:OJ 的客户端，作为前端运行，可以进行单个文件提交（输出界面较友好）、多个文件提交（以 json 形式输出）、比赛列表查看（以表格形式输出）、新建用户（方便测试调试）；

⑦、file_analysis.rs:负责有关于文件的处理，包括将全局变量读入文件，从文件中读出全局变量，文件内容的清理更新等。

三、**主要功能：**

①、通过 post/jobs 发送 json 形式的文件，来提交一次答案，后端接收任务并进行题目查找，各测试点测试，最后返回任务编号和响应结果，评测中支持多种语言的输入，不合法的语言会在命令行 Command 编译错误后返回 Err 错误，在每个测试点评判时，standard 会忽略文件末空行和行末空格，而 strict 不会，特殊的 misc 会有特殊处理；

②、通过发送 get/jobs 的请求，来获取已经提交的 jobs 列表；

③、通过发送 get/jobs/{jobId}的请求，来获取指定编号的任务，不存在会 404 NOT FOUND；

④、通过发送 put/jobs/{jobId}的请求，会重新评测某一任务，返回新结果，并且更新文件存储下该任务的数据；

⑤、通过发送 post/users，并且附带一个 json 格式内容，如果用户 ID 为 None 且姓名未出现重名，则是新建用户，否则为按编号修改用户名，并更改文件存储下的全部数据。出现无效 ID 和重名现象会返回 Bad Request 或 Not Found 等；

⑥、发送 get/users，会获得当前的全部用户列表，默认初始的用户为{id:0,name:”root”}

⑦、发送 post/contests 检查合法性后，会在总配置基础上，新创建一场比赛；

⑧、发送 get/contests，会获得目前全部的比赛列表；

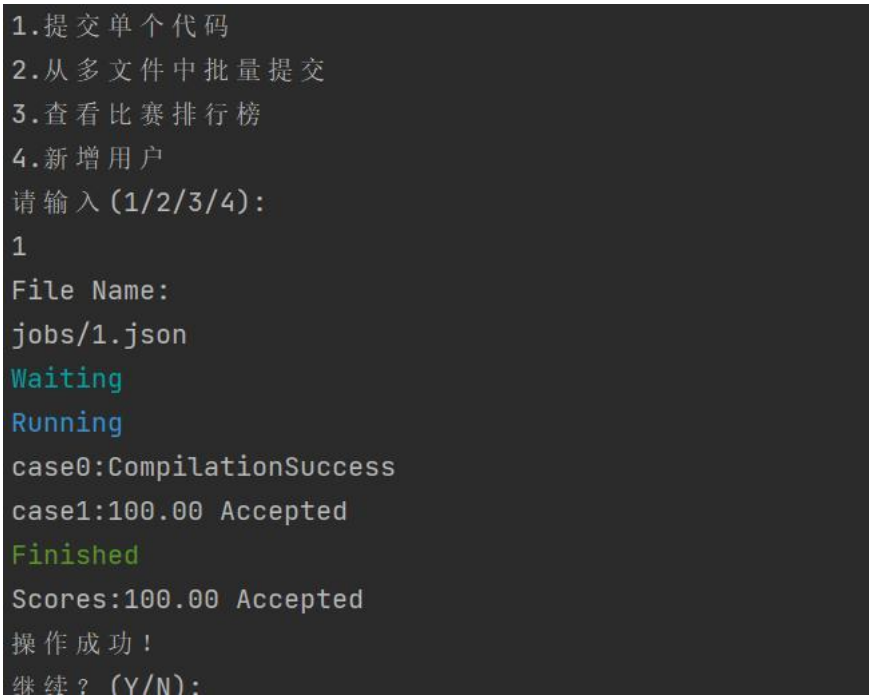
⑨、发送 get/contests/{contestId}，会获得指定 id 的比赛数据，未找到会返回 Not Found，

如果 id<0 会有报错，无法解析为 usize；

⑩、get/contests/{contestId}/ranklist，会获得某场比赛的用户排行榜，排行榜的排行规则将由 scoring_rule 和 tie_breaker 来决定，算法中，优先级更高的排序方式会更晚被 sort，例如优先分数，其次 id，则先通过 id 来 sort，再用 score 来 sort，tie_breaker 下无法打破平局会赋予同名次，算法为，从第二名起，如果他和前一人平局，赋予其与前一人相同名次。

四、提高功能：

①、Cli 前端，可以通过前端来实现单个文件提交、多个文件提交、查看某比赛排行榜、注册用户的操作；



指令 1，提交单个文件

```
操作成功！
继续？(Y/N):
y
1.提交单个代码
2.从多文件中批量提交
3.查看比赛排行榜
4.新增用户
请输入(1/2/3/4):
2
File Name:
jobs/1.json
File Name:
jobs/2.json
File Name:

[
  {
    "id": 2,
    "created_time": "2022-09-09T13:43:35.191Z",
    "updated_time": "2022-09-09T13:43:35.789Z",
    "submission": {
      "source_code": "fn main(){\n    println!(\"Hello World!\");\r
```

指令 2，提交多个文件，并以 json 形式返回

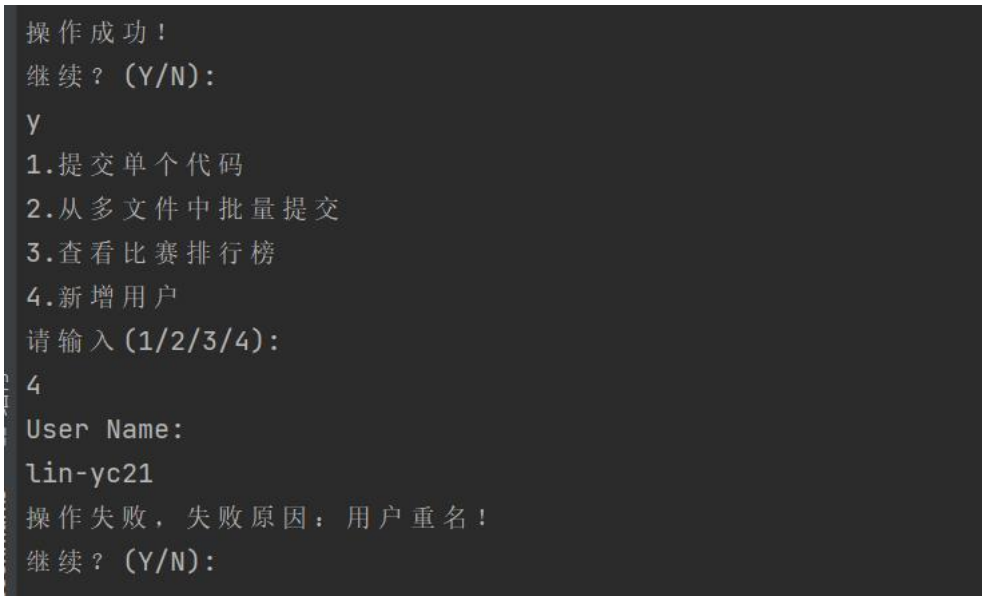
```
操作成功！
继续？(Y/N):
y
1.提交单个代码
2.从多文件中批量提交
3.查看比赛排行榜
4.新增用户
请输入(1/2/3/4):
3
Contest ID:
0
Rank  ID   Name        Scores
001   000  root        [100.0, 0.0, 0.0]
001   002  user2       [100.0, 0.0, 0.0]
003   001  lin-yc21    [0.0, 0.0, 0.0]
003   003  userx       [0.0, 0.0, 0.0]
003   004  woo         [0.0, 0.0, 0.0]
操作成功！
继续？(Y/N):
```

指令 3，查看排行榜

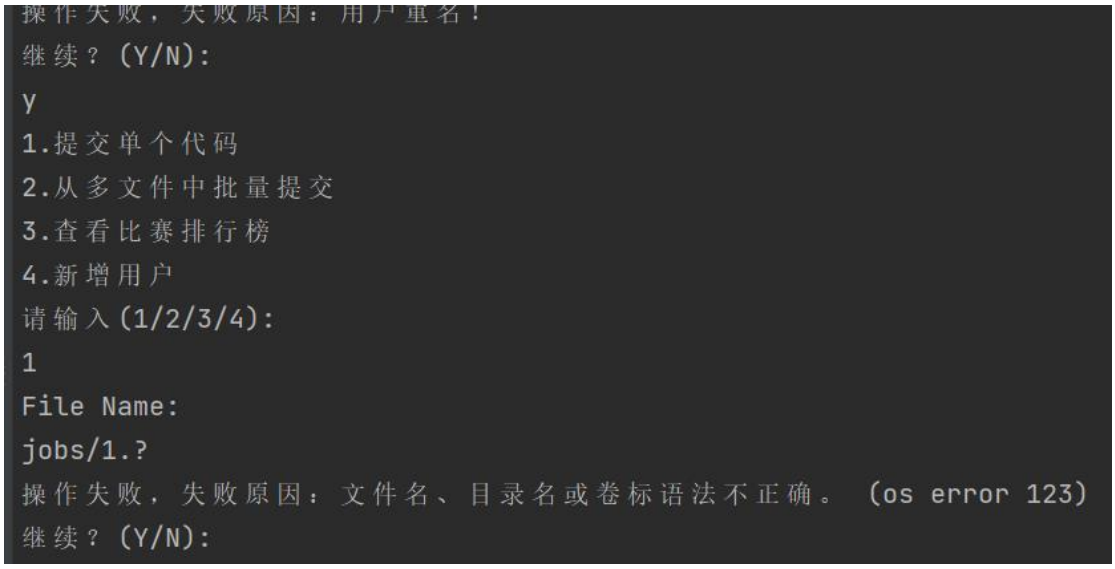
```
1.提交单个代码
2.从多文件中批量提交
3.查看比赛排行榜
4.新增用户
请输入(1/2/3/4):
4
User Name:
woo
操作成功！
继续？(Y/N):
z
```

Version Control　TODO　问题　终端　Python Packages　服

指令 4，注册用户，方便测试 cli 前端



报错 1，用户重名



报错 2，文件不合法

Cli 前端主要依赖于 reqwest::client 来向后端发送请求，得到相应响应 reqwest::Response，并且将该响应以 json 形式进行解析，向命令行显示相关参数；

Cli 和 oj 都属于 bin 文件夹的下两个 binary，可在两个终端同时运行。

②、多比赛支持，可以支持 contest_id>0 比赛信息的录入，比赛的信息会存入到 CONTEST_ARRAY 中，并且会对之后的各类任务进行有关比赛编号的筛选，对应/contests 的一系列功能。

③、持久化存储(json)，将所有的全局静态变量信息在每局开始时读取，结束时保存，并且实时更新（即在每次诸如 post 等操作时就更新并保存文件内容，在 get 时会从文件中读取并更新，全局静态变量的作用是方便各个程序块的调用和提高读取效率；

④、非阻塞评测，在 post/jobs 中实现，每次发送该请求会立即返回 queuing 的状态，并且把任务放入到新开辟的 block 新线程中，利用 api_terminal 函数进行处理，在处理发生时会在 QUEUEING_ARRAY 中放置该任务，状态为 running，当任务处理结束后，会放置到 RESPONSE_ARRAY 中，状态变为 finished。因此 get/jobs 会获取到 queuing，running 和 finished

三种状态。

⑤、资源限制进阶，利用每个任务在其单独开辟的文件夹内所占用的字节数大小，来计算其内存，并且在内存占用过大时返回报错

⑥、Special_judge，如果 problem 的 misc 中有 spj，则会忽略默认的检测方式，采用 spj 中提供的命令行参数来进行解析和检测，并以此作为评判的标准；如果 spj 的命令行参数有误，则会返回 SPJ ERROR。

⑦、竞争得分，如果 problem 的 misc 中有 dynamic_ranking_ratio，则每个人的获取分数为正确性+时间竞争得分，如果某一点发生错误，只能拿到正确性得分，题目的每一个测试点都有最短用时，每个人每个测试点的竞争时间得分都为竞争分*最短时/个人时。

五、无意义的感想：

其实还是蛮高兴能够独立完成一个规模较大的大作业，相比上次感觉要说的话反而没那么多了，认识到自己的不足，仍需不懈的努力，感谢助教的答疑和同学的鼓励，我承认自己是不够优秀不够自信，但是我有毅力和耐心走完整个旅程，下一门语言，我准备好了。