2024-2025 学年第二学期 程序设计实习 大作业报告
暨 PKU Offline Judge 使用手册

#46 赛博判官队 赵泽宇 鄢宇阳 丁羽珩

一、程序编译与运行需求

由于此程序使用了 Windows API 进行进程间通信以及获取进程运行 CPU 时间和虚拟内存占用，故须在 Windows 系统上运行。运行此项目前，请确保已经安装 MINGW 编译器和 python3 解释器，并将其路径添加到系统环境变量 PATH 中，以确保评测编译阶段的顺利运行。

如果你希望从源代码编译此项目，请遵循以下要求：

1. 确保 Qt 版本≥6.9.0，并安装对应的 QtPDF 组件。
2. 在项目构建 make 阶段添加参数 install，然后构建项目。（或手动复制 testlib.h 到构建目录下）
3. 运行 windeployqt6.exe 补全动态链接库及资源文件。

二、程序功能及使用方法介绍

2.1 程序功能

本项目是集出题、考试、评测、评分于一体的离线评测系统，支持 C/C++及 Python 三种编程语言，全面适配广泛使用的 testlib.h 评测工具库（by Mike Mirzanyanov），并可评测多种题型：

- 提交答案题
- 传统代码题
- 编程填空题（支持多空）
- NOI 式交互题
- 进程间通信式交互题
- 通信题

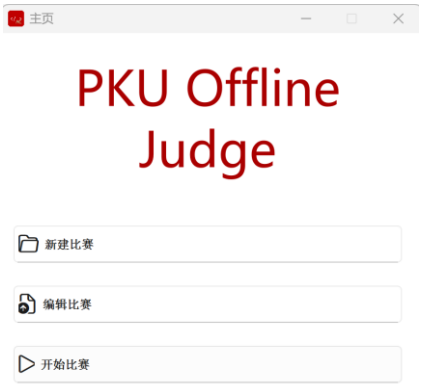
目前网络上可评测进程间通信式交互题的工具均为需要部署在 Linux 服务器上的在线 OJ 系统，体积庞大，部署繁琐；支持测评通信题的 OJ 则只有 UOJ 等在造题流程上自由度较大的 OJ；对于本地的评测工具 Cena、Lemon 等则不支持进程间通信式题目的评测，且没有确认考生身份的提交界面。因此，本项目虽仍有一些不足之处，但在填补本地评测工具的空白上有重要意义。

本程序实现考试流程的方式如下：

1. 教师修改项目内密匙后编译项目，该密匙将用于加密学生试题和提交文件包。
2. 教师在创建/编辑比赛页面填写比赛基本信息及导入学生账号密码，对每道题目上传题干 PDF 文件和评测需要的文件，设置编译流程、测试点及子任务信息、数据生成及验证流程、评测信息，并运行测试点生成及验证流程。（记得手动点击保存按钮）
3. 所有题目测试数据均验证完成后，可导出加密的学生试题包文件。学生在考试界面导入试题包，输入正确的账号密码后方可登录考试系统，待比赛开始后方可进入考试界面看题及答题；学生登录和考试界面均显示账号，方便核对考生身份。
4. 考试界面内，考生可随时查看试题，但仅可在考试进行期间发起测评。学生提交的代码仅会在被设置为样例的测试点上评测，并显示结果和测试点得分。所有测试点测试完成后（不论结果，此举是为了防止学生以恶意代码攻击评测系统致使计算机关机等），方可提交此题代码。学生可在任意时刻导出其提交的所有代码的加密文件包，供教师端测评。
5. 教师端在编辑比赛-评测面板内导入学生加密代码包，执行一键评测，并可导出所有学生成绩的 CSV 文件包。

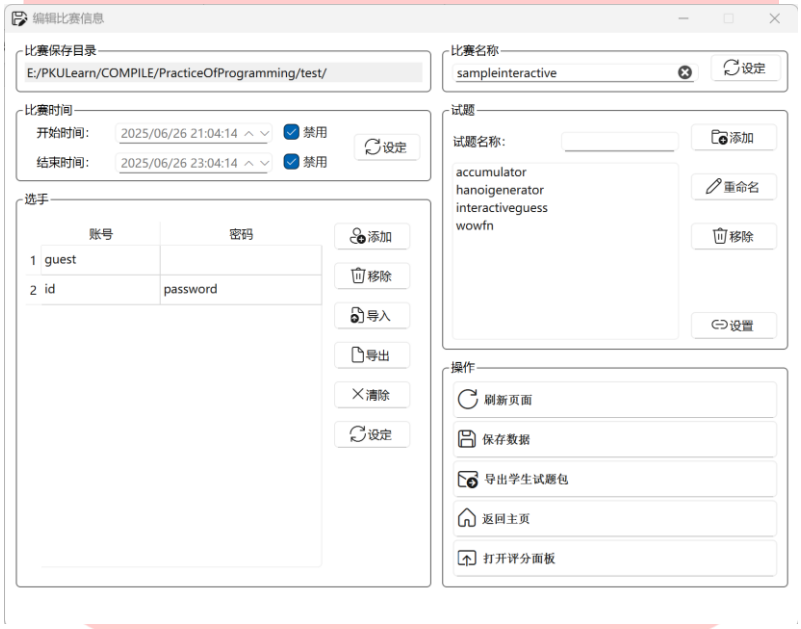
2.2 程序界面【此部分包含程序的使用说明，同时提供示例文件】

2.2.1 主界面



教师点击“新建比赛”可创建新的试题包；点击“编辑比赛”并选择对应的.ctinfo 文件可自动解包储存的文件并进入比赛编辑界面；学生点击“开始比赛”并选择对应的.sctinfo 文件（加密试题包）可进入登录界面。

2.2.2 比赛编辑器



每项功能的说明都已清晰标注在页面上，有几点说明：

1. 选手账号密码导入/导出采用.csv 格式，没有标题行，每行格式为**账号,密码**；
2. 所有选项点击**设定**按钮后才会被修改；退出本页面前务必记得点击**保存数据**按钮！
3. 比赛及试题名称要求只包含数字、字母和_；账号只允许包含数字、字母和_.@；密码只允许包含数字、字母和_.@~%$#!*；不符合以上要求的内容将导致设置失败；
4. 点击刷新会重置页面为已设定的状态，未按**设定**按钮的改动将被丢弃。

2.2.3 试题编辑器

关于评测资源的说明：

文件名只可包含字母、数字和._，包含以下类别：

内置文件-代码：仅有 testlib.h，不可添加，不可删除，不可修改

资源文件-代码/文本：上传文件，全过程有效。区别在于代码可作为编译的输入文件，文本不可以。

测试数据-代码：数据验证器、生成器、标答，这些文件仅在数据生成及验证阶段有效，不会打包进学生试题包。

测试数据-文本：所有测试点，每个测试点名称为文件名+测试点编号+后缀名（如有），仅样例测试点会被打包给学生。

选手提交文件-代码：需要提交的代码

选手提交文件-代码模板/代码片段：编程填空题的程序主题，需填空的部位用<snippet filename="代码片段文件名.snip">标记，文件后缀为.c.tpl/.cpp.tpl/.py.tpl 等，程序会自动解析须填空的部位，将这些部位作为提交文件-代码片段呈现，并在评测时生成去掉.tpl 的文件进行编译和执行。在界面上将显示不含.tpl 的文件名，但储存的文件和输入的文件名须含.tpl。

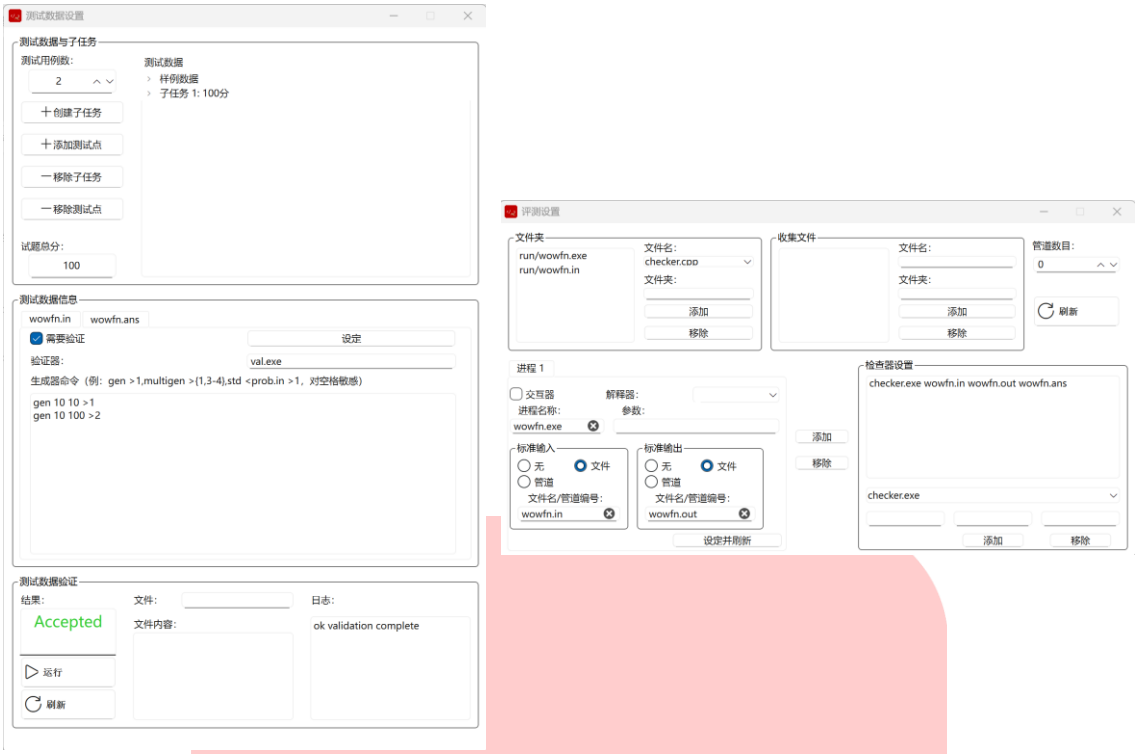
选手提交文件-文本：提交答案题使用，对于每个测试点，选手需提交名称为文件名+测试点编号+后缀名（如有）的文件。

关于编译设置的说明：

预编译阶段指在数据生成及验证前的编译阶段（不能编译选手提交文件）；评测前阶段指在选手提交了文件准备测试前的编译阶段（不能编译测试数据-代码文件）（需注意的是：评测前编译文件会编译所有输入文件不含测试数据-代码类别文件的条目，并非只是属于评测前阶段的条目）

关于返回：关闭此页面即返回比赛设置页面

2.2.4 测试数据编辑器和评测流程编辑器



关于子任务的说明：每个测试点的得分为 0~1 之间的实数，子任务得分为子任务中测试点最低分乘上子任务分值；问题得分为子任务得分之和。

关于 Testlib 使用的说明：生成器、验证器、交互器、检查器均需使用与 testlib 相同的输入输出及参数表格式，因为评测结果是根据返回值以及读取标准错误流输出的结果共同判定的。其中验证器、检查器必须为使用 C++编写并编译得到的文件，交互器、生成器可以使用 python。相关格式请参阅 TestlibInfo.md，这里不再赘述。注意：对于部分分，如果采用 partially correct，输出得分应为满分为 200 的整数；如果采用 points，输出应为 0~1 之间的实数。

使用标程生成答案：std <prob.in >caseid 实际上运行的是 std，输入：probcaseid.in，输出：probcaseid.ans。

数据生成成功，各测试点齐全且需验证的测试点均验证通过，则数据生成及验证流程通过，否则为不通过，不能导出学生试题包。

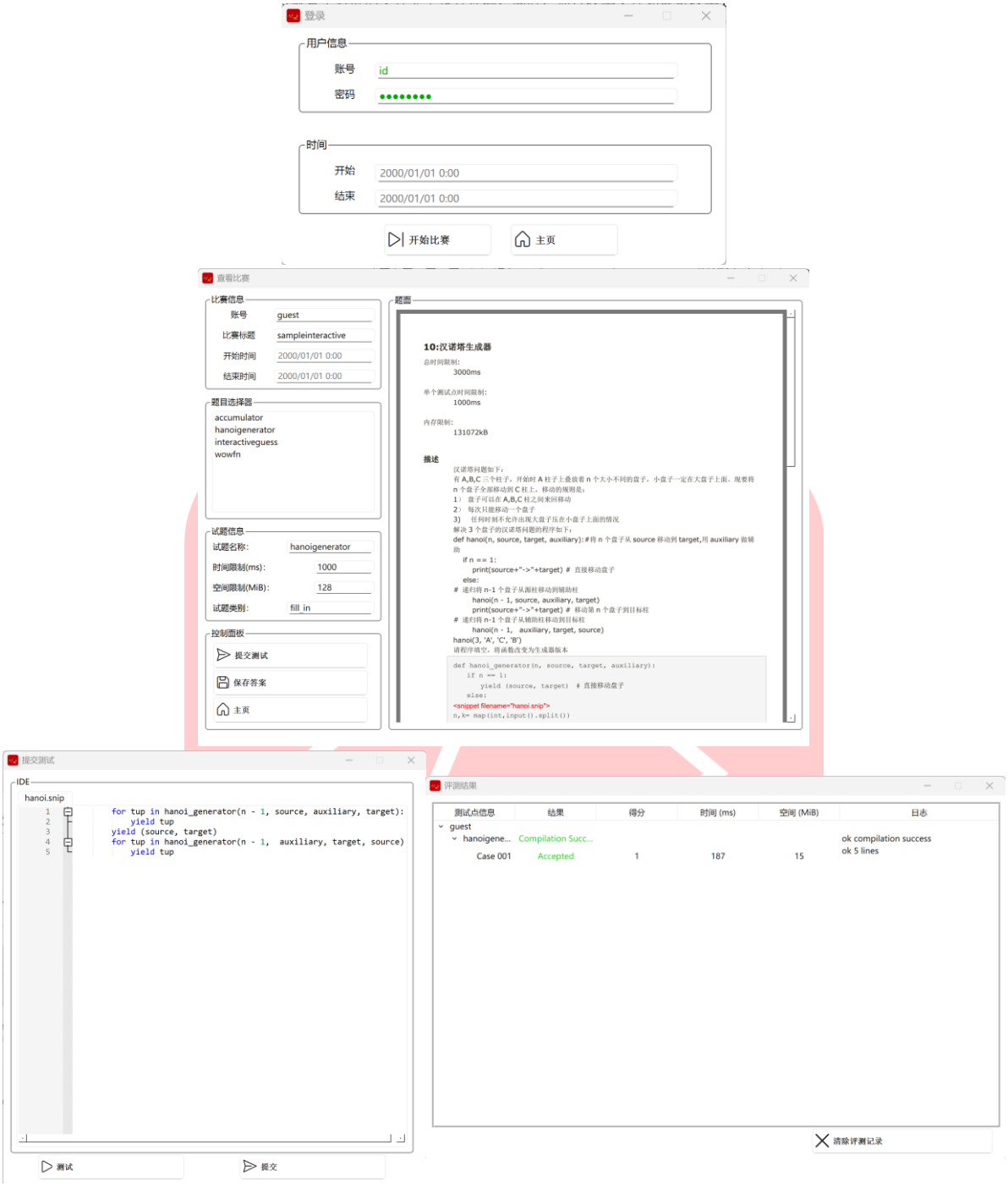
关于管道：管道编号为 0~管道数-1，相同管道编号的输入输出会连接在一起，请给每个管道恰好分配一个输入和一个输出进程。

关于文件：设置输入/输出文件后，对应文件将自动链接到程序的标准输入输出，如果希望程序自己执行输入输出重定向，此栏请设置为“无”。

文件夹/收集文件：这是为了防止学生程序试图访问当前测试点答案作弊。学生程序和对应的输入文件将被置于对应的文件夹下执行评测，同时，通过收集文件功能可收集处于文件夹内的程序输出文件及交互器输出文件。注意：如果选择将文件输出链接到对应程序，无需收集对应的文件。

关于返回：关闭此页面即返回比赛设置页面

2.2.5 学生端

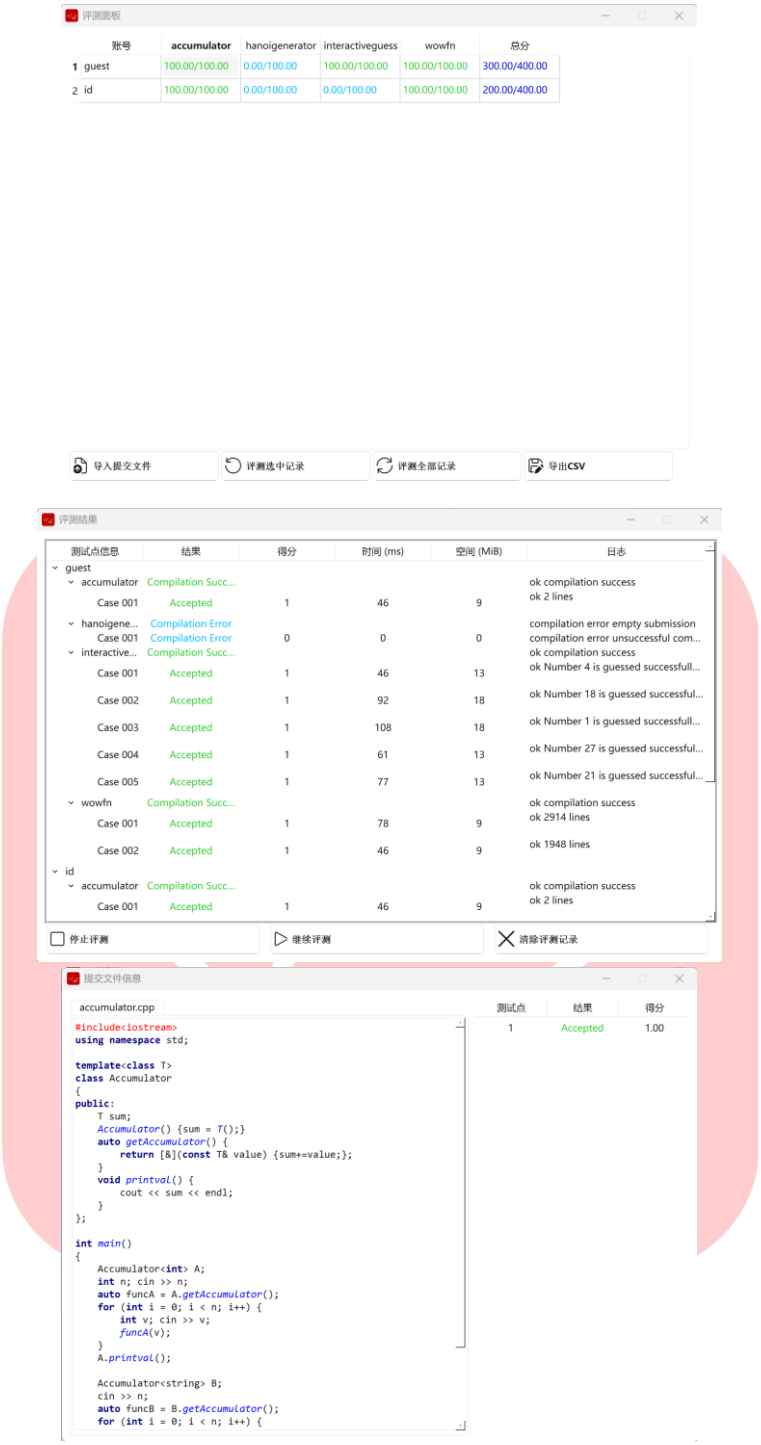


代码提交**测试**后可以看到评测结果，双击评测结果条目可以看到完整的日志。

学生端代码提交页面提供使用 Qscintilla 实现的简易编辑器，支持行号显示、语法高亮、自动缩进、代码折叠、括号匹配以及对于代码内定义的变量、函数的自动补全。（但不支持库函数自动补全）

只有在规定比赛时间内完成**测试**（无论结果）的代码才可以**提交**，**提交**后的代码才会在**保存答案**时被打包。

2.2.6 教师评测



导入.sspack 文件，且经用户名、密码、试题名称核验通过，提交方为有效，会被评测机评测。选中部分列名为题目名称的格子，可只评测对应的提交记录。双击格子可以看到各测试点得分和提交的代码。

重要提示：如果系统资源压力过大，可能会出现编译超时、程序无法启动或启动超时的情况，此时，对应测试点将会得到 FAIL 的结果，请重测对应记录。

三、项目构成模块介绍及工作量统计【注：受篇幅限制，不在此处放源代码】

总体规范：尽可能将后端功能与前端分离，数据储存与数据处理分离，并使不同页面之间功能尽可能独立，从而便于调试；方法名、变量名、类名直接体现用途，从而无需额外注释。

3.1 使用的外源项目

3.1.1 testlib by Mike Mirzanyanov

便于造题的评测库 under MIT License

3.1.2 QAESEncryption

AES 加密库 under Unlicense

3.1.3 QScintilla by Riverbank Computing

IDE 控件库 under GPLv3 License

3.2 核心功能类

3.2.1 ctsettings.h

存放了核心的数据存储、处理、加密解密等功能，其中包括：

- enum TResult、DescriptionStr、DescriptionCol、parseVerdict：处理评测结果储存、解析、输出
- CompileOp：数据验证及评测时执行编译并获取编译结果
- keyStr：密匙；Encryption：用于加密、解密的工具类，采用 QAESEncryption
- StrVal：验证字符串格式的工具类
- FolderOp、FileOp：处理文件/文件夹复制、移动、删除及文件全文读取、写入的工具类
- Codetpl、get_filename_with_id、parseCombinedArgString：解析代码模板、生成带测试点 ID 文件名、解析命令行参数的工具
- ctinfo、sctinfo、sspack、templ：文件后缀名常量
- JudgeProcess、Testdata、Problem、Contest：比赛数据储存类，这些类均配备了打包为 JSON 和从 JSON 解析的函数，Contest 还配备了打包文件数据的函数。所有文件均以 QByteArray 读取并以 Base64 储存。
- JudgeInfo：选手提交评测的文件，兼有记录评测结果的功能；可通过 packInfoList、getInfoList 批量打包学生在不同题目的所有提交。

3.2.2 procexeclib.h

存放使用 WinAPI 实现的核心评测运行库，包括：

- THandle：内核安全的句柄，会在析构时自动关闭句柄
- TPipe：自动构建管道句柄
- TFile：快捷构建文件句柄
- TProcess：进程创建、重定向、启动、终止及时间内存监测类
- THandleDevice：对句柄进行读写的 QIODevice 派生类，用于读取交互器错误流输出。

3.2.3 val_utils.h

存放数据生成及验证流程工具：

- GenOp：运行数据生成流程并返回生成结果
- ValidateOp：运行数据验证流程并返回生成结果

3.2.4 judge_utils.h

- JudgeOp：运行评测并获取总时间、空间消耗及评测结果和得分（交互器）
- CheckOp：运行检查器获取评测结果和得分

3.2.5 texthighlighter.h/cpp

TextHighlighter 类：派生自 QSyntaxHighlighter 的 QPlainTextEdit 语法高亮处理器，用于处理教师端查看学生提交的代码时对代码的语法高亮。

3.2.6 qcodeedit.h/cpp

QCodeEdit 类：派生自 QsciScintilla 的代码编辑器组件，用于学生端提交，

3.3 工具窗口类

3.3.1 dataconfigwidget.h/cpp/ui

测试数据编辑器的子窗体，用于单个测试点生成、验证流程的设置

3.3.2 procexecinfowidget.h/cpp/ui

评测流程设置的子窗体，负责评测时单个进程运行和重定向的设置

3.3.3 judgingwidget.h/cpp/ui

存放测评机实现窗体和接口，评测线程与窗体运行时分离，并作了线程安全处理。

- JudgingThread：分离式评测线程，主要为了防止评测时出现 UI 界面卡死的情况。此窗体内部使用 QList 储存评测流程，其中编译和评测的信息储存与执行通过 JudgeRunner 类派生出的 cplRunner 和 judRunner 利用虚函数 StartJud 的多态实现，评测开始和评测结果传输利用信号和槽机制实现。
- JudgingWidget：评测控制面板与显示器

3.3.4 submissioninfo.h/cpp/ui

教师端评测时显示学生代码及测试点得分窗体

3.3.5 login.h/cpp/ui

学生端登录界面，同时负责在登录成功后解包相关文件

3.4 功能窗口类

3.4.1 mainwindow.h/cpp/ui

主页，同时负责在打开比赛文件时解包比赛试题包文件

3.4.2 contesteditor.h/cpp/ui

比赛信息编辑页面，同时作为其他设置窗体和评测面板的入口以及学生包导出的入口

3.4.3 judgepanel.h/cpp/ui

教师端评测页面，支持学生提交批量导入，一键测评，部分题重测，查看学生代码和导出结果

3.4.4 problemeditor.h/cpp/ui

问题信息编辑页面

3.4.5 testdataprocessor.h/cpp/ui

测试数据信息编辑页面及测试数据生成、验证执行工具

3.4.6 judgesetting.h/cpp/ui

评测流程设置页面

3.4.7 studenteditor.h/cpp/ui

学生端题目查看、代码测试及提交包导出

3.5 杂项

3.5.1 main.cpp

Qt 自动生成的程序启动入口

3.5.2 ProjectOJ_zh_CN.ts

使用 QtLinguist 编辑的翻译文件

四、项目分工信息

4.1 总览

赵泽宇：负责与造题、答题全流程设计、实现，以及测评机核心实现相关的部分

鄢宇阳：负责与学生端打包、身份核验、代码提交相关的部分

丁羽珩：负责评测流程数据获取界面与储存，以及 IDE 的相关部分

4.2 各部分分工概览

4.2.1 程序（注：代码行数为.h 和.cpp 文件行数之和）

文件名	工作量（行）	参与情况			文件名	工作量（行）	参与情况		
		赵泽宇	鄢宇阳	丁羽珩			赵泽宇	鄢宇阳	丁羽珩
ctsettings	1209	√	√	√	submissioninfo	85	√		
procexeclib	374	√			login	130		√	
val_utils	126	√			mainwindow	148	√	√	
judge_utils	225	√			contesteditor	452	√	√	
texthighlighter	170			√	judgepanel	327	√		
qcodeedit	228	√		√	problemeditor	764	√		√
dataconfigwidget	194	√			testdataprocessor	374	√		
procexecinfowidget	116	√			judgesetting	265			√
judgingwidget	604	√			studenteditor	199	√	√	

总计：5990 行

4.2.2 材料

内容	参与情况		
	赵泽宇	鄢宇阳	丁羽珩
功能设计文档	√		
初版录屏	√	√	
终版录屏			√
TestlibInfo.md	√		√
ProjectOJ_zh_CN.ts	√		
sample	√		
作业报告	√	√	√

五、项目总结与反思

5.1 项目总结

本项目实现了基于 windows 系统的上机考试系统，且支持多种题型的评测，总体功能完善，且对于进程间通信式交互题、通信题、编程填空题的支持是同类型离线评测系统所不支持的。此外，本项目依托于 Qt 实现了较现代的界面风格，并利用 QScintilla 实现了简易的学生端代码编辑器，界面美观整洁。

5.2 不足之处

作为首次开展团队项目的尝试，在此项目的开发过程中我们遇到了许多问题。

其一是团队分工不够明确，工作流程安排不够合理，导致有些时候某些已实现的功能需要等待其他功能实现后才能开展测试。

其二是时间把控不够好，向项目内添加了过多且实现技术难度较大的功能。虽然这些功能最终得以实现，但导致项目完成时间严重延后，未能在路演前完成部分功能。

其三是项目向仓库提交不够及时，导致代码合并花费大量时间。

其四是评测对系统资源要求高，系统资源压力较大时可能会出现进程启动超时的现象，但总体上出现此情况的概率较低。

5.3 展望

以下功能受限于时间和技术原因未能实现，但可能在未来成为项目的一部分：

1. 添加对于更多编程语言的支持
2. 简化对于预设试题类型的造题流程
3. 在沙箱中评测，以彻底解决选手攻击测评机的问题