

Guidance

Full Name	Student ID
-----------	------------

Yuchen Wang	U202141XXX
-------------	------------

Under the MIT license.

Contents

1	Introduction	1
2	Usage	2
2.1	OJ Structure	2
2.2	Testbench	2
3	References	3
A	Upload docker container to CG platform	4
B	title in toc	5

1 | Introduction

To fill up the blank of verilog online judger, which could boost the verifying of open-problem in teaching and in exam, we designed this verilog judger based on `iverilog`, `python` and `bash`. It can be simply run by using a docker container in CG platform, or somewhere else, and generate a `json` output via standard output for further usage.

It could automatically detect the number of test points, and offers the reference and wrong answer of the first test point where error occurs.

2 | Usage

2.1 | OJ Structure

The structure of the core which belongs to the Verilog OJ is shown in Figure 2.1.

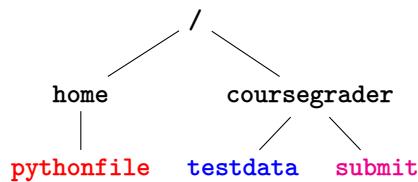


Figure 2.1: File structure

Where `pythonfile` contains python and shell scripts, it is the core of the judge; the represented structure of `testdata` is shown in Figure 2.2, `pointi`, which represents the sub-testbench, is optional¹, and how many sub-testbenches depends on the number of test points n in the `testdata`; student's answer stores in `submit`, where should contain `*.v` and NOT contain `*_tb.v`.

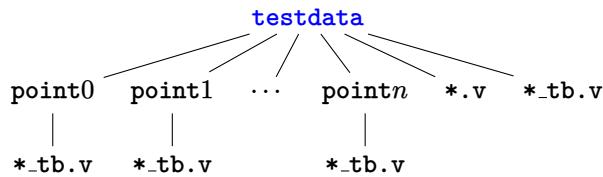


Figure 2.2: File structure of `testdata`

2.2 | Testbench

Testbench should offer some output through the console, otherwise it can not judge whether the answer is right or wrong. Sentences such as `$display`, `$monitor` or `$write` can be used.

¹If it doesn't pass the main testbench, sub-testbench will work. Otherwise, those sub-testbench will be bypassed.

3 | References

A | Upload docker container to CG platform

CG 平台 Docker 使用流程

VANITY FAIR^a

^a University of Science and Technology Beijing, 30 XUEYUAN ROAD, HAIDIAN DISTRICT, Beijing, 100083, Beijing, China

Abstract

本文介绍了如何在 CG 平台上使用、部署 Docker，并给出了目前 Verilog、Java 测评机的使用文档。

Keywords: CG, Docker

目录

1 CG 平台挂载 Docker

1

1. CG 平台挂载 Docker

由于尝试使用 Dockerfile 在线构建镜像失败（无响应），因此本文采用自行开发 Docker 镜像的方式部署 Docker 测评机。

首先需要以系统管理员身份进入系统管理 → 实验环境管理 → 桌面镜像管理或 Jupyter 镜像。

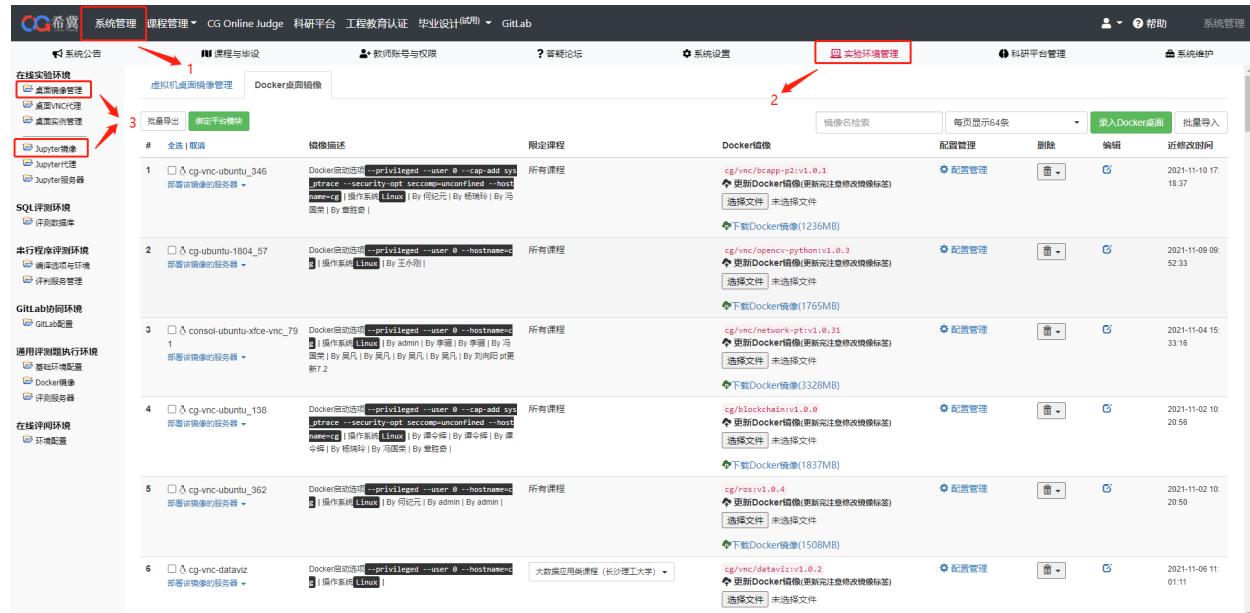


图 1: 进入镜像管理界面

之后点击批量导入，导入镜像文件

The top screenshot shows the 'Docker桌面镜像' (Docker Desktop Image) management page. It lists several Docker containers with their descriptions, status, and actions. A red box highlights the '批量导入' (Batch Import) button at the top right of the table header.

The bottom screenshot shows a 'Import Docker Image' dialog box. It has two main sections: 'Import Docker Image' where a file named 'dockerimages.1.tgz' is selected, and a preview section showing the uploaded image details. Red arrows point to the 'Browse...' button in the dialog and the 'Import Docker Image' button in the background table.

图 2: 导入镜像文件

点击浏览，导入 Docker 镜像

图 3: 导入 Docker 镜像

等待镜像上传完毕（镜像上传过程中此页面不要进行其他操作，不可同时上传多个镜像）

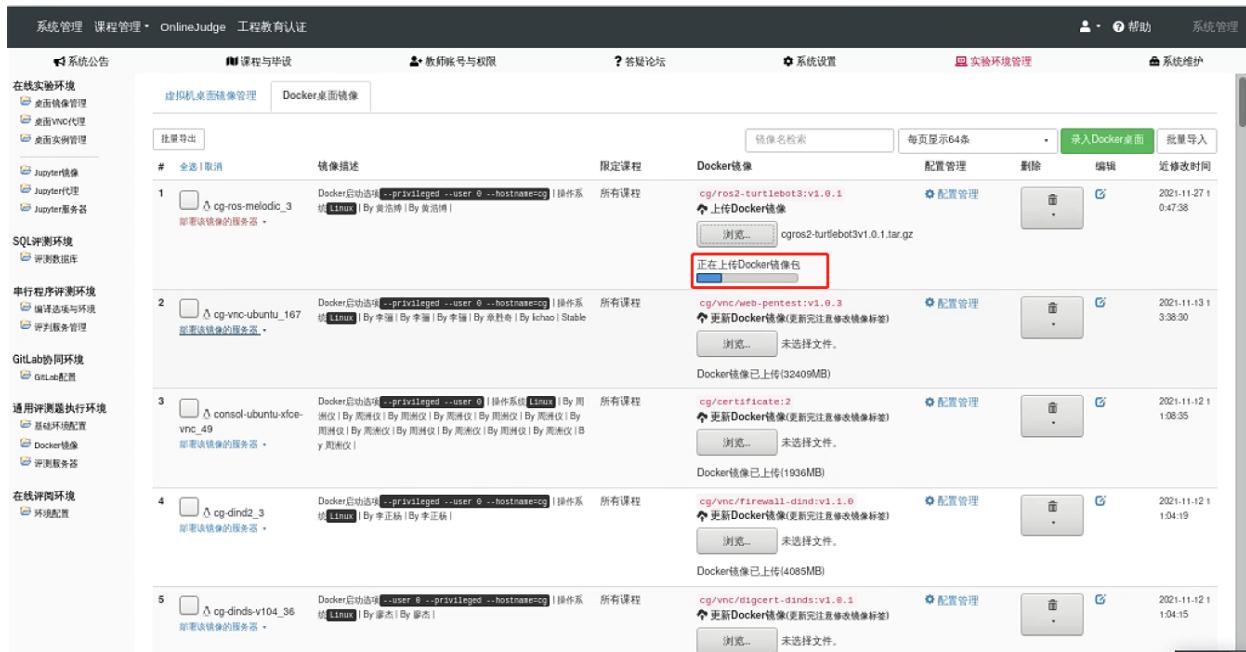


图 4: 等待镜像上传完毕

点击桌面实例管理或 Jupyter 服务器，找到部署镜像的服务器点击编辑

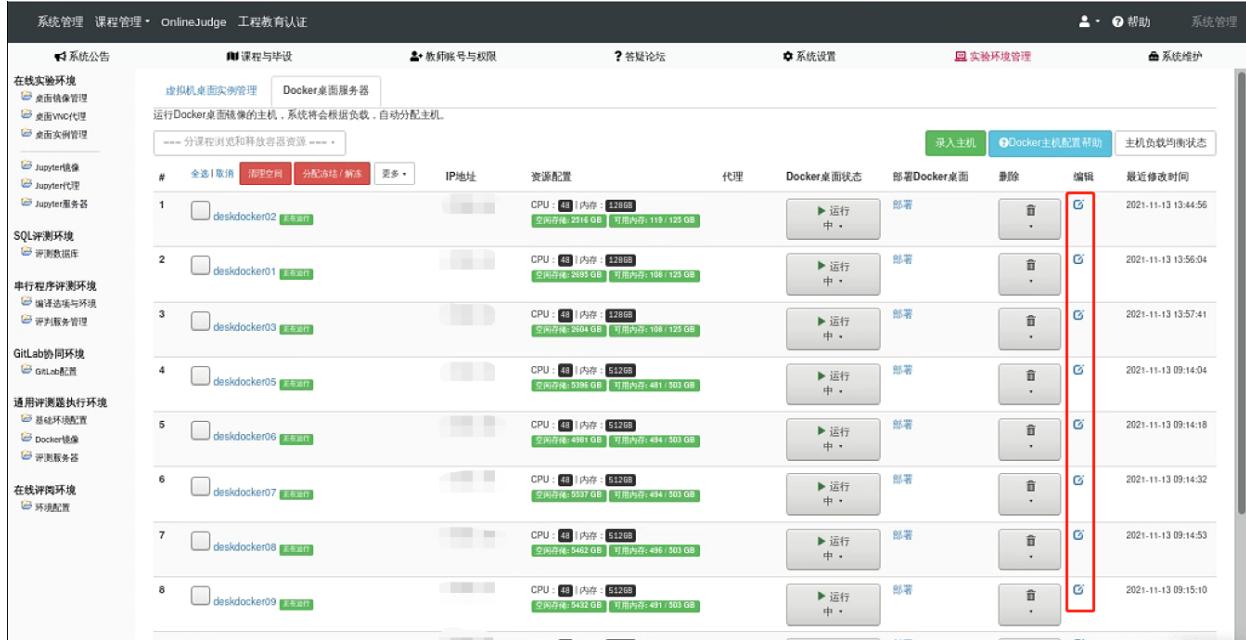


图 5: 编辑服务器

将需要部署的镜像加入到“需部署的镜像”中（单击左键勾选）

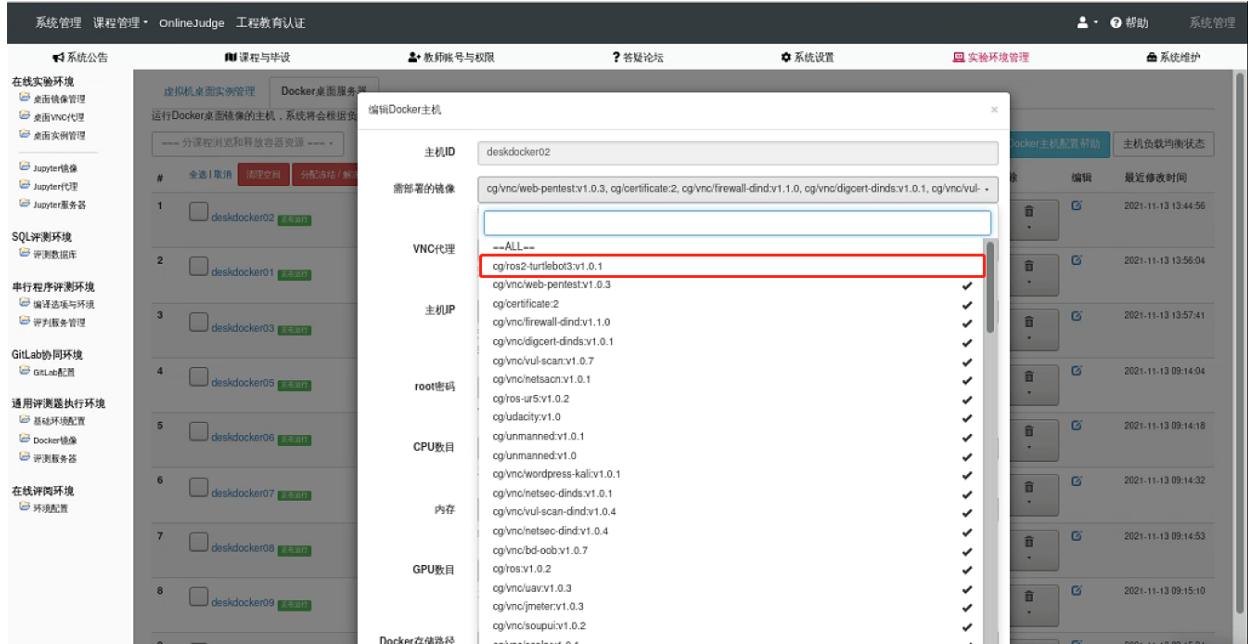


图 6: 加入镜像

点击保存



图 7: 保存镜像

服务器 Docker 桌面状态将改为黄色，并提示镜像部署状态异常，点击部署

The screenshot shows a list of Docker desktop instances. The first instance, deskdocker02, has its deployment status highlighted with a red box. The deployment status for deskdocker02 is labeled '部署' (Deployed). Other instances show deployment statuses like '运行中' (Running) or '未部署' (Not Deployed).

#	IP地址	资源配置	代理	Docker桌面状态	部署	删除	编辑	最近修改时间
1	172.29.1.94	CPU: 48 内存: 128GB 空闲内存: 2016 GB 可用内存: 119 / 125 GB		▶ 运行中	部署			2021-11-27 10:56:38
2	172.29.1.93	CPU: 48 内存: 128GB 空闲内存: 2095 GB 可用内存: 106 / 125 GB		▶ 运行中	部署			2021-11-13 13:56:04
3	172.29.1.95	CPU: 48 内存: 128GB 空闲内存: 2004 GB 可用内存: 106 / 125 GB		▶ 运行中	部署			2021-11-13 13:57:41
4	172.29.1.99	CPU: 48 内存: 128GB 空闲内存: 2036 GB 可用内存: 481 / 503 GB		▶ 运行中	部署			2021-11-13 09:14:04
5	172.29.1.100	CPU: 48 内存: 128GB 空闲内存: 4981 GB 可用内存: 494 / 503 GB		▶ 运行中	部署			2021-11-13 09:14:18
6	172.29.1.101	CPU: 48 内存: 128GB 空闲内存: 2037 GB 可用内存: 494 / 503 GB		▶ 运行中	部署			2021-11-13 09:14:32
7	172.29.1.108	CPU: 48 内存: 128GB 空闲内存: 5462 GB 可用内存: 494 / 503 GB		▶ 运行中	部署			2021-11-13 09:14:53
8	172.29.1.97	CPU: 48 内存: 128GB 空闲内存: 5432 GB 可用内存: 491 / 503 GB		▶ 运行中	部署			2021-11-13 09:15:10

图 8: 部署状态异常

点击重新部署，将会拉取镜像（拉取镜像过程中页面可进行其他操作）

The screenshot shows a modal dialog box for redeployment. The dialog box contains the text: "正在拉取镜像: cg/ROS2-turtlebot3:v1.0.1, 请耐心等待, 可以关闭本页面稍后再来查看。" (Pulling image: cg/ROS2-turtlebot3:v1.0.1, please wait patiently, you can close this page and check it later.)

图 9: 重新部署

再次点击重新部署

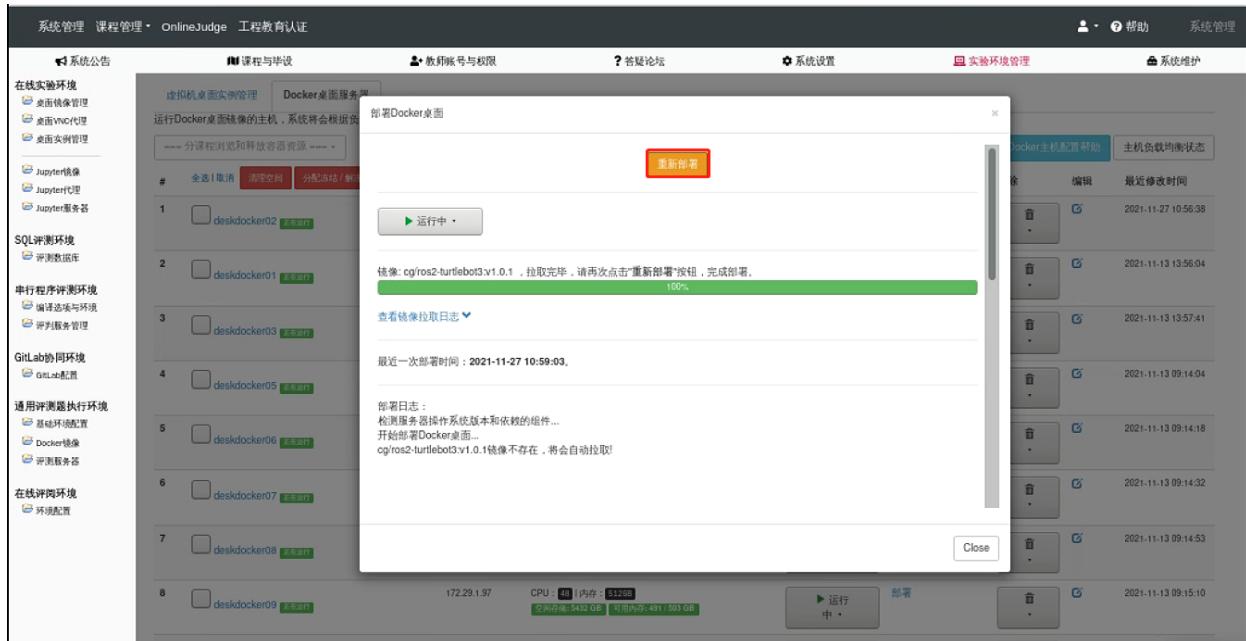


图 10: 再次重新部署

出现此界面即镜像部署完成

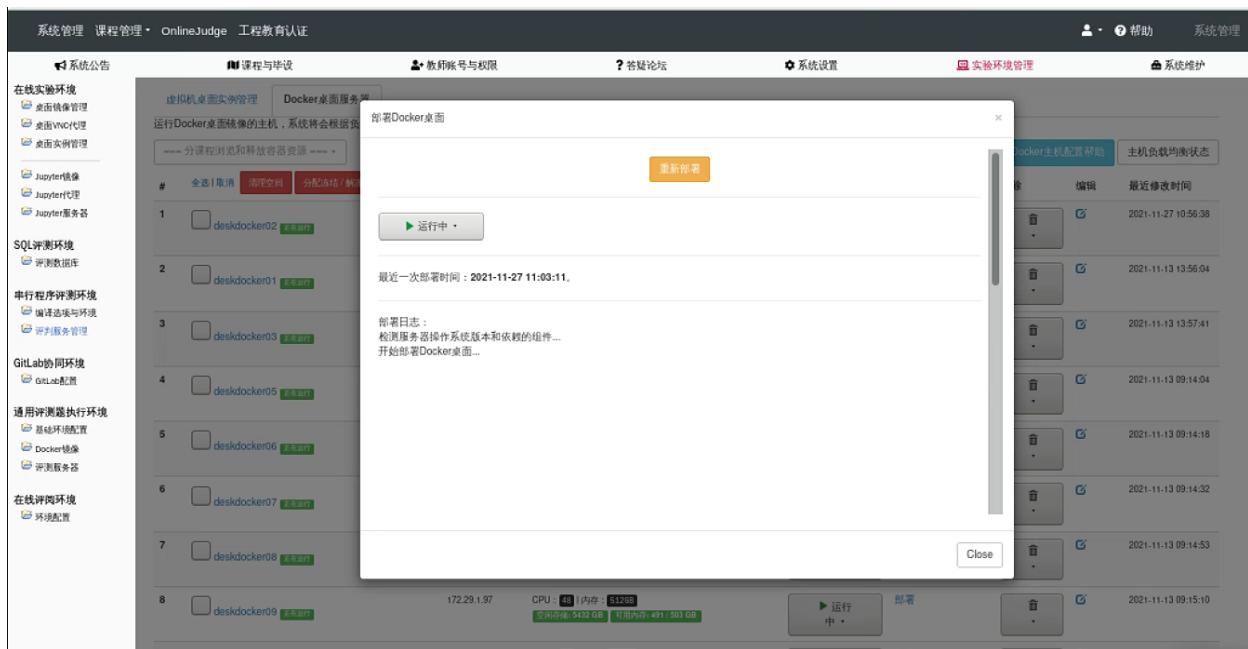


图 11: 部署完成

2. 添加通用评测题

进入作业栏目，单击录入题目中的添加通用评测题，并在运行环境中选择相应评测机。

The figure consists of two vertically stacked screenshots of a web-based assignment creation tool.

Top Screenshot (Basic Information and Evaluation Constraints):

- Left Sidebar:** Lists various assignment types: 单选题 (Single Choice), 选择题 (Multiple Choice), 填空题 (Fill-in-the-blanks), 判断题 (True/False), 简答题 (Short Answer), 拍照上传题 (Photo Upload), 编程题 (Programming), 程序片段编程题 (Program Segment Programming), 接口编程题 (Interface Programming), SQL 评估题 (SQL Evaluation), 算法可视化 (Algorithm Visualization), 文件上传题 (File Upload), 技能题 (Skill), 通用评测题 (General Test Task) (highlighted with a red box), Web 布尔题 (Web Boolean), 和 Scratch 编程题.
- Header:** Includes course settings, student and teacher management, exam, cloud lab, online classroom, help, and a log-in student link.
- Content Area:**
 - 基本信息 (Basic Information):** Contains fields for 题目标题 (Title), 题目内容 (Content), and 元素路径 (Element Path).
 - 属性信息 (Attribute Information):** Includes fields for 难度 (Difficulty), 知识点 (Knowledge Points), and 章节 (Chapter). Buttons for creating new knowledge points and detailed sections are present.
 - 评判约束 (Evaluation Constraints):** Includes fields for 最长运行时间 (Max Runtime), 内存限制 (Memory Limit), and 运行环境 (Runtime Environment). A dropdown menu for selecting the runtime environment is shown expanded, listing options like judge/java, usbtologinsjudgenv/latest, usbt/judge/v1.2, test/judge/v5, etc.

Bottom Screenshot (Correctness Verification):

- Left Sidebar:** Same as the top screenshot.
- Header:** Same as the top screenshot.
- Content Area:**
 - 评判结果 (Evaluation Results):** Offers options for 得分 (Score) and 排行榜 (Leaderboard). A note says '得分是最为常见的情形，提交之后直接给出分数。排行榜常见的场景是性能优化化，基于排行榜进行评分，查看排行榜的详细解释'.
 - 正确性验证 (Correctness Verification):** Contains a 'Test Data Editor' tab (highlighted with a red box) and a 'Help' tab.
 - 在线开发环境 (Online Development Environment):** Shows a dropdown for 选择在线开发环境 (Select Online Development Environment) and a '保存' (Save) button.

图 12: 添加通用评测题

在正确性验证中打开测试数据编辑器

我的课程

公告与问卷

作业题库

项目概览

录入题目

单选题

选择题

填空题

判断题

简答题

拍照上传题

编程题

程序片段编程题

接口编程题

SQL评测题

算法可视化

文件上传题

接龙题

通用评测题

Web集成题

Scratch编程题

定制题型

通用评测工作台

回收站

作业管理

作业概览

布置作业

回收站

快速入门

最长执行时间 2 秒
如果执行时间超过限制，将会强行退出

内存限制 1024 MB
如果执行时占用内存超过此限制，将会强行退出

运行环境

代码提交方式 第二文件提交 在线编辑
提交并运行：直接通过代码，然后提交评测。支持项目级多源文件的提交； 在线编辑：在浏览器内填写代码并提交，支持预览代码框架。
注意不要在学生提交过程中切换界面模式，会造成提交答案丢失！

提交文件 大小不超过 10 KB 文件后缀限定 rar, zip
提交文件大小必须固定，不能小于 0 KB；文件后缀为空表示不限定。如有多个后缀，逗号分隔，例如 .rar, .zip, .java, rar和zip压缩包，会自动解压，然后挂载到Docker执行环境。

评判规则 ① 得分 ② 排行榜
得分 是最常有的情形，提交之后直接给出分数。 排行榜 常见的场景是性能优化，基于排行榜进行评分，查看排行榜的详细解释

正确性验证 测试数据 打开测试数据编辑器 帮助
测试数据将会以只读方式挂载到评测Docker容器上。

在线开发环境 开发环境 ==选择在线开发环境==
非必选项，为每个学生或者小组配置一个在线桌面或者Web IDE，方便学生在线开发。

保存 202.204.62.160/admin/courseAdmin/testCaseEditor/uploadEditor.jsp?testEditorID=437251

图 13: 打开测试数据编辑器

在测试数据中选择 File 中的 Open

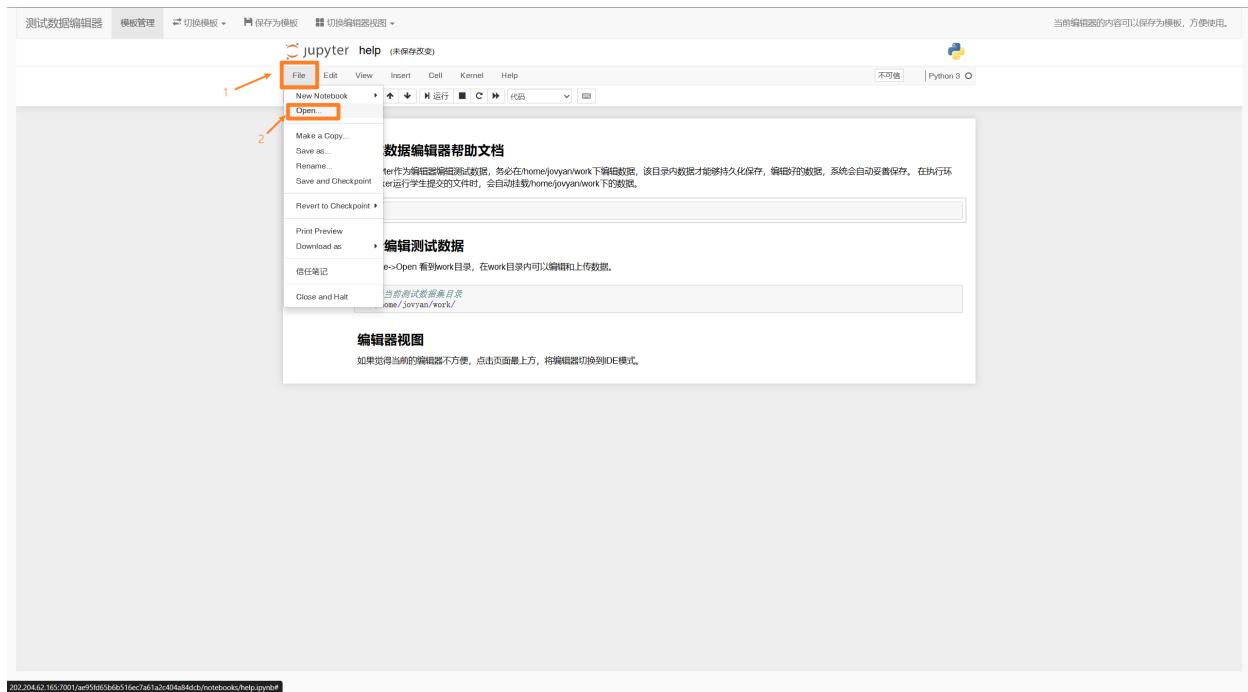


图 14: 打开测试数据

进入 work 文件夹，选择 Upload 即可上传数据



图 15: 上传数据