2023-2024年重庆市职业院校技能大赛

区块链技术应用赛项规程

一、赛项信息

<table>
<tr><td colspan="4">赛项组别</td></tr>
<tr><td colspan="4">☐中等职业教育☑高等职业教育</td></tr>
<tr><td colspan="4">☑学生赛（☐个人/☑团体）☐师生同赛 ☐教师赛（☐个人/☐团体）</td></tr>
<tr><td colspan="4">涉及专业大类、专业类、专业及核心课程</td></tr>
<tr><td>专业大类</td><td>专业类</td><td>专业名称</td><td>核心课程
（对应每个专业，明确涉及的专业核心课程）</td></tr>
<tr><td rowspan="11">电子信息</td><td rowspan="11">计算机类</td><td>510212
区块链技术应用</td><td>区块链核心技术
虚拟化及容器技术
区块链部署与运维
区块链应用设计与开发智能合约开发</td></tr>
<tr><td>510202
计算机网络技术</td><td>Linux 操作系统管理
网络自动化运维</td></tr>
<tr><td>510201
计算机应用技术</td><td>前端设计与开发
系统部署与运维</td></tr>
<tr><td>510203 软件技术</td><td>软件建模与设计
软件测试</td></tr>
<tr><td>510205 大数据技术</td><td>大数据平台部署与运维</td></tr>
<tr><td>510206
云计算技术应用</td><td>容器云服务架构与运维</td></tr>
<tr><td>510207
信息安全技术应用</td><td>操作系统安全
信息安全产品配置与应用电子数据取证技术应用</td></tr>
<tr><td>510208
虚拟现实技术应用</td><td>软硬件系统搭建和维护</td></tr>
<tr><td>510209
人工智能技术应用</td><td>人工智能系统部署与运维</td></tr>
<tr><td>510213
移动应用开发</td><td>面向对象建模与设计
服务端框架技术</td></tr>
<tr><td>510216
密码技术应用</td><td>信息安全技术与实施
公钥基础设施应用
信息安全工程与管理</td></tr>
</table>

二、竞赛目标

党的二十大提出了“强化国家战略科技力量、坚决打赢关键核心技术攻坚战”战略部署，国家“十四五”规划提出了“加快推动数字产业化”要求。区块链作为新兴数字产业，在产品溯源、数据流通、供应链管理等领域具有广泛的应用前景，在推动国家经济体系实现技术变革、重构数字产业体系中发挥了重要作用。

区块链技术应用赛项围绕区块链技术在产业应用中的工作岗位技能要求而设计，赛项内容覆盖区块链产业主流的技术方向。通过大赛培养参赛选手在企业真实项目环境下进行区块链平台框架搭建、区块链产品需求分析与方案设计、区块链系统部署、区块链系统运维与监测、智能合约开发、区块链应用软件前端与后端开发、区块链系统测试及调优等方面的能力，形成良好的职业素养，全面提升学生自主解决综合问题的能力，达到“以赛促教、以赛促学、以赛促改、赛课融通、赛训结合”目的。

通过赛项的设置，加强职业院校与区块链产业的衔接，引导院校与企业共同开发区块链课程和资源，促进教师开展区块链关键应用技术研究，推进区块链技术技能人才培养，实现产业链、创新链与教育链协同创新，促进产教融合和科教融汇，服务国家“网络强国、数字中国”战略。

三、竞赛内容

（一）选手需具备的能力

区块链技术应用赛项根据《全国职业院校技能大赛执行规划（2023—2027 年）》，结合高职区块链技术应用专业简介，针对区块链新兴数字产业所需的技术技能，面向区块链应用开发、智能合约开发、区块链测试、区块链运维、区块链运营等岗位，区块链应用

设计与开发、区块链平台部署与运维、智能合约开发与测试、区块链应用软件开发等典型工作任务，基于企业实际项目，要求选手在规定时间内完成指定任务的区块链应用开发。赛项主要考查选手对区块链系统应用需求分析与方案设计，区块链应用及智能合约设计与开发，区块链系统测试设计、执行与分析，区块链系统部署、维护和监控，基于区块链系统的应用软件前端与后端开发等专业核心能力及职业素养，全面检验学生在区块链技术应用的工程实践能力和创新能力，展现高职区块链人才培养成果。竞赛技能要求及成绩比例见表 1。

表 1 竞赛技能要求及成绩比例

竞赛内容	技能要求	成绩比例
区块链系统部署与运维	1.熟悉Linux操作系统运维 2.熟练使用Docker容器 3.熟悉常见的区块链技术架构及运行机制，能搭建和配置区块链平台及网络 4.熟练使用MySQL等主流数据库，完成业务系统数据库的创建和管理 5.熟练使用脚本、编程语言和日志分析工具快速定位问题 6.熟悉区块链访问接口、数据格式，能快速分析区块链节点状态和系统运行状态	25%
区块链系统测试	1.具备测试需求分析及测试用例设计能力 2.熟练使用常见的测试工具 3.能对区块链系统进行防篡改测试和签名测试 4.具备全链路压力测试设计和执行能力	10%
智能合约开发	1.熟悉Solidity基本语法 2.具备智能合约编程的能力 3.具备智能合约部署和调用的能力	20%
智能合约测试	1.熟悉智能合约的运行机制 2.具备智能合约测试的能力	10%

区块链应用 前端开发	1.熟悉常用前端开发技术 2.具备UI设计能力 3.具备页面逻辑编程能力	10%
区块链应用 后端开发	1.熟悉Java等后端开发语言及框架技术 2.具备后端编程调用智能合约能力 3.具备后端编程访问数据库能力	20%
职业素养	1.具有良好的文档写作能力，代码编写规范 2.具有团队合作精神和创新意识	5%

（二）竞赛模块

表 2 竞赛模块内容

模块		主要内容	比赛时长	分值
模块一	区块链产品方案设计与系统运维	根据项目背景描述完成区块链产品的需求分析与方案设计，在Linux环境下完成区块链系统的部署、运维及测试	90min	35分
模块二	智能合约开发与测试	根据给定的区块链业务需求编写功能需求文档和智能合约设计文档；使用Solidity编程语言开发智能合约，设计符合需求的合约接口，完成合约功能的开发，对智能合约进行编译、部署和调用；进行智能合约测试	75min	30分
模块三	区块链应用系统开发	利用前端开发语言及框架完成页面逻辑设计和展示；利用Java等后端开发语言及框架，实现应用程序接口，完善区块链应用系统，调用智能合约实现链上信息的查询和结果展示	75min	30分
职业素养		团队分工合理、操作规范、文明竞赛		5分
合计			240min	100分

1.模块一：区块链网络部署

（1）基于给定的环境和区块链系统，完成系统部署及节点部署。使用监控工具完成对网络、节点服务的监控。根据业务需求，完成系统日志、网络参数、节点服务等系统结构的维护。

（2）设计区块链系统的测试流程，调用智能合约进行单元测试、集成测试、系统测试和性能测试；根据业务需求，分析并修复给定智能合约中的安全漏洞。

2.模块二：智能合约开发与测试

（1）使用Solidity语言进行智能合约开发，完成智能合约的部署和调用。

（2）编写智能合约单元测试代码并完成智能合约的功能测试和性能测试。

3.模块三：区块链应用系统开发

（1）根据业务需求，使用前端开发框架完成页面设计，使用已提供的服务端接口获取业务数据，并进行部署与展示。

（2）依据功能需求，使用Java等后端开发语言及常用框架进行后端代码开发，访问数据库、实现应用程序接口、调用智能合约，完善区块链应用系统，完成后端代码的部署。

4.职业素养

要求参赛选手文档写作科学规范，具有团队合作精神和创新意识，比赛操作严谨，代码编写规范，文明竞赛。

四、竞赛方式

（一）竞赛形式

线下比赛。

（二）组队方式

1.竞赛以团体赛方式进行，以院校为单位参赛，每支参赛队由 2 名选手组成，不得跨校组队，同一学校参赛队不超过 2 队。参赛队可配指导教师，指导教师须为本校专兼职教师，每队限报 2 名指导

教师，竞赛期间不允许指导教师进入赛场进行现场指导。指导教师负责参赛选手的报名、训练指导、服务和比赛期间参赛选手的日常管理。

2.参赛选手须为高等职业院校专科、高等职业院校本科全日制在籍学生（以报名时的学籍信息为准）、五年制高职（四、五年级）在籍注册学生。凡在往届全国职业院校技能大赛中获一等奖的选手，不能再参加今年同一专业类赛项的比赛。

五、竞赛流程

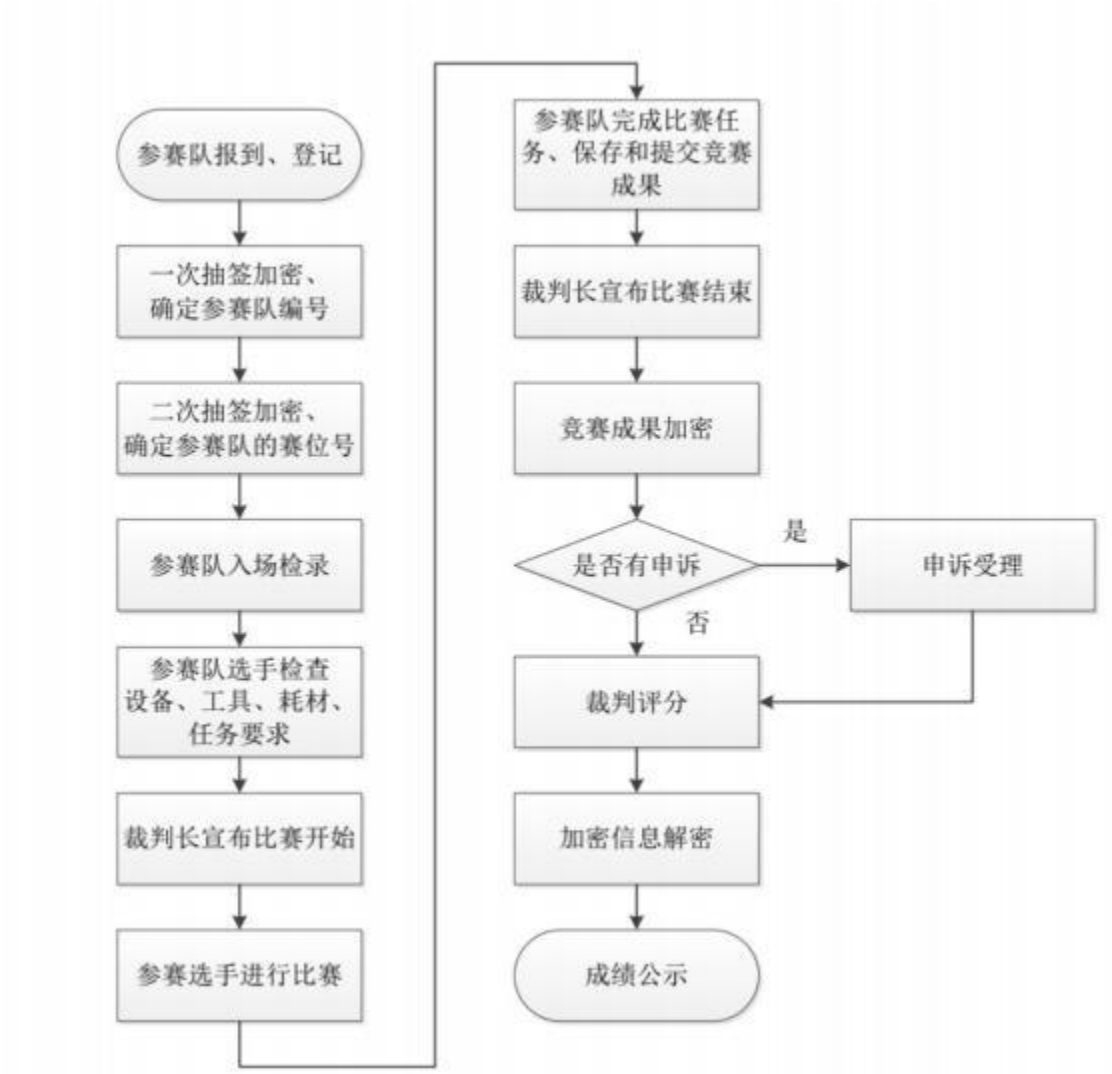
（一）竞赛日程表

竞赛报到地点在重庆电子工程职业学院南校区 4 栋 105 会议室，比赛地点在重庆电子工程职业学院南校区 4 栋三楼，具体日程安排见表 3 所示。

表 3 竞赛日程表

日期	时间	内容	参加人员
报到日	09:00-14:00	参赛队报到，领取资料	工作人员、参赛队
	13:00-14:00	赛前工作会议	专家组、裁判组、监督仲裁
	14:00-15:00	裁判培训会议	专家组、裁判组
	15:00-16:00	领队会，抽签确定检录顺序号	各参赛队领队、裁判长
	16:00-16:40	熟悉赛场	各参赛队领队
	17:00	检查封闭赛场	裁判长
竞赛日	07:45	参赛队到达竞赛场地前集合	各参赛队、工作人员
	07:45-08:45	1）大赛检录 2）工位号抽签	1）参赛选手、检录裁判 2）参赛选手、加密裁判
	08:45-09:00	环境确认	参赛选手、现场裁判
	09:00-13:00	实操比赛	参赛选手
	14:00-18:30	现场评分、抽检和统分	裁判
	19:00-21:00	成绩核定及公示	裁判长、评分裁判、监督仲裁

（二）竞赛流程图



六、竞赛规则

1.大赛以学校为单位组织报名参赛，各参赛校按照大赛确定的报名时间和名额，通过重庆市职业院校技能大赛组委会办公室（以下简称“重庆市大赛办”）公布的报名网址进行报名并完成相关审批工作，地址：http://125.84.158.248:8085

2.参赛队可于正式比赛 1 天前，由主办方统一组织熟悉场地。比赛赛位通过抽签决定，参赛选手在赛前 10 分钟领取比赛任务，并进入比赛赛位，比赛正式开始后方可进行相关操作。比赛期间参赛选手原则上不得离开比赛场地。选手分工、工作程序和时间安排由参赛队自行决定。

4.竞赛所需的硬件、软件和辅助工具统一提供，参赛队不得使用自带的任何具有存储和通信功能的设备。

5.竞赛过程中，选手须严格遵守操作规程，确保人身及设备安全，并接受裁判员的监督和警示。选手如有疑问，应举手示意，现场裁判应按要求及时予以答疑。如遇设备或软件故障，参赛选手应举手示意，现场裁判、技术人员等应及时予以解决。若因非参赛选手个人因素造成设备或软件故障，由裁判长视具体情况做出裁决。若因选手因素造成设备故障或损坏，无法继续竞赛，裁判长有权决定终止该队竞赛。

6.竞赛结束后，参赛队要确认已成功提交所有竞赛文档，裁判员与参赛队队长一起签字确认，参赛队在确认后不得再进行任何操作。

7.最终竞赛成绩经复核无误及裁判长、监督仲裁长签字确认后，在指定地点，以纸质形式向全体参赛队进行公布，同时成绩上报重庆市大赛办，获奖情况由大赛办统一公布。

8.赛项每个比赛环节的裁判判分原始材料和最终成绩等结果性材料经监督仲裁组人员和裁判长签字后装袋密封留档，上交重庆市大赛办。

七、技术规范

依照《全国职业院校技能大赛赛项规程编制要求》，结合企业职业岗位对人才培养需求，并参照相关国家职业标准制定。参赛代表队在实施竞赛项目中要求遵循的规范见表 4。

表 4 竞赛模块内容

序号	标准号	内容
1	GB/T11457-2006	信息技术、软件工程术语
2	LD/T81.1-2006	职业技能实训和鉴定设备技术规范
3	GB/T25069-2010	信息安全技术术语
4	ISO22739-2020	Block chain and distributed ledger technologies—Vocabulary（区块链和分布式账本技术词汇）
5	GBZ2-02-10-15	区块链工程技术人员国家职业技术技能标准
6	GBZ4-04-05-06	区块链应用操作员国家职业技术技能标准
7	CBD-Forum-001-2017	区块链参考架构
8	CBD-Forum-002-2017	区块链数据格式规范
9	T/SIA007-2018	区块链平台基础技术要求
10	CIET-2018-04	区块链技术人才培养标准

八、技术环境

（一）竞赛环境

1.竞赛场地：竞赛场地分为竞赛现场、裁判员休息区、指导老师休息区，以上区域应保证采光、照明和通风良好。竞赛现场配置录像设备，实时录制赛场内竞赛情况。

2.竞赛设备：场内竞赛区按照参赛队数量准备比赛所需的软硬件平台，为参赛队提供统一竞赛设备和备用设备，选手无需自带任何工具及材料。

3.竞赛工位：竞赛现场每个参赛队工作区间面积合理，确保参赛队之间互不干扰，每个参赛队工作区上标明编号。

4.竞赛现场符合消防安全规定，现场消防器材和消防栓合格有效，应急照明设施状态合格，赛场明显张贴紧急疏散图，赛场地面

张贴疏散指示箭头，赛场出入口专人负责，随时保证安全通道的畅通无阻。

（二）竞赛设备及软件

表 5 硬件设备

设备名称	备注
服务器	高性能工作站 处理器i7/64G内存/固态硬盘1T及以上、USB3.0接口、千兆及以上网卡
交换机	二、三层可控交换机（千兆、带电源）
PC	通用台式机 处理器 i5/32G 内存/固态硬盘 512G 及以上、USB3.0 接口、千兆及以上网卡

表 6 服务端软件

序号	软件	备注
1	区块链平台	国产主流的区块链平台及其管理工具

表 7 PC 工具软件

序号	名称	版本	用途
1	WPS	Version2019及以上	编制文档
2	Putty	Version0.7	远程工具
3	WinSCP	Version5.15及以上	文件传输
4	Chrome	Version100及以上	浏览器
5	JavaJDK	Version11及以上	Java开发
6	Visual Studio Code	Version1.75及以上	开发工具
7	ItelliJIDEA	VersionCE2023	开发工具
8	Gradle	Version6及以上	构建工具

9	Maven	Version3.6及以上	构建工具
10	Office Visio	Version2013及以上	绘图工具
11	MySQL	Version8.0及以上	数据库工具
12	Vue	Version2.x	前端框架
13	Postman	Version9.0及以上	接口测试工具
14	竞赛管理平台	Version1.0及以上	竞赛管理

九、竞赛样卷

（一）赛题内容

1.模块一：区块链网络部署

（1）基于给定的环境和区块链系统，完成系统部署及节点部署。使用监控工具完成对网络、节点服务的监控。根据业务需求，完成系统日志、网络参数、节点服务等系统结构的维护。

（2）设计区块链系统的测试流程，调用智能合约进行单元测试、集成测试、系统测试和性能测试；根据业务需求，分析并修复给定智能合约中的安全漏洞。

2.模块二：智能合约开发与测试

（1）使用Solidity语言进行智能合约开发，完成智能合约的部署和调用。

（2）编写智能合约单元测试代码并完成智能合约的功能测试和性能测试。

3.模块三：区块链应用系统开发

（1）根据业务需求，使用前端开发框架完成页面设计，使用已提供的服务端接口获取业务数据，并进行部署与展示。

（2）依据功能需求，使用Java等后端开发语言及常用框架进行后端代码开发，访问数据库、实现应用程序接口、调用智能合约，完善区块链应用系统，完成后端代码的部署。

（二）竞赛样卷

样卷见附件1。

十、赛项安全

1.赛前组织专人对比赛现场、住宿场所和交通保障进行考察，并对安全工作提出明确要求。赛场的布置，赛场内的器材、设备，应符合国家有关安全规定。承办院校赛前须排除安全隐患。

2.赛场周围要设立警戒线，防止无关人员进入，发生意外事件。在具有危险性的操作环节，裁判员要严防选手出现错误操作。

3.制定开放赛场和体验区的人员疏导方案。赛场环境中如存在人员密集、车流与人流交错的区域，除了设置齐全的指示标志外，须增加引导人员，并开辟备用通道。

4.大赛期间，承办院校须在赛场设置医疗医护工作站。在管理的关键岗位，增加力量，建立安全管理日志。

5.参赛选手、赛项裁判、工作人员严禁携带通讯、摄录设备和未经许可的记录用具进入比赛区域；如确有需要，由赛项承办单位统一配置，统一管理。赛项可根据需要配置安检设备，对进入赛场重要区域的人员进行安检，可在赛场相关区域安放无线屏蔽设备。

6.比赛期间发生意外事故时，发现者应在第一时间报告重庆市大赛办，同时采取措施，避免事态扩大。事后赛项承办校应向重庆市大赛办报告详细情况。

十一、成绩评定

（一）评分原则

1.竞赛评分严格遵守公开、公平、公正、独立、透明的原则，区块链技术应用赛项评分采用赛项结果评分方法，赛项最终得分按百分制计算，贯彻落实大赛坚持的公平、公正和公开原则。

2.赛项合作企业不得直接或者间接地参与赛项评分。

3.赛项评分依据选手固化在实操任务中的成果，通过评分裁判对比赛成果再现的方法评分，并兼顾团队协作精神和职业素养综合评定。

4.为了确保赛事评判的客观性，制定详细的评分标准，细化评分项目，尽可能量化每一评分项目的评分标准，减少主观判断比例，确保赛事客观公正。

5.评分过程全程可追溯。

（二）评分标准

表8 评分标准

赛项模块	竞赛内容	考核的知识点、技能点	相应得分点	分值
模块一 区块链产品方案设计与系统运维	区块链产品需求分析与方案设计	依据给定的项目背景，分析业务需求，编制业务流程图、活动图、类图、时序图、系统结构图、系统架构图等，编写项目概要设计说明书，完成产品原型及软件功能的设计	1.掌握区块链系统基本设计概念，合理划分角色及业务功能 2.文档编制规范，各模型图绘制正确 3.模块及功能划分完整、合理 4.正确撰写应用系统功能设计文档	10
	区块链系统部署与运维	基于给定的环境和区块链系统，完成区块链系统部署及节点部署。通过监控工具完成对网络、节点服务的监控。根据业务需求规范，完成系统日志、网络参数、节点服务等系统结构的维护	1.按要求正确部署区块链网络,并能验证运行状态 2.正确安装管理工具及监控工具 3.正确完成业务系统数据库的创建和管理 4.搭建的区块链符合业务	15

			需求，按要求进行扩容和网络配置等维护操作	
	区块链系统测试	设计区块链系统的测试流程，调用智能合约进行单元测试、集成测试、系统测试和性能测试；根据业务需求，分析并修复给定智能合约中的安全漏洞	1.对测试需求分析正确，合理设计测试用例 2.正确对区块链系统进行防篡改测试、签名测试等 3.正确使用测试工具修复合约中的漏洞 4.正确对已部署的智能合约进行性能测试、系统测试和执行分析	10
模块二 智能合约 开发与测试	智能合约设计	根据区块链业务需求，编写功能需求文档和智能合约设计文档	1.合约模块划分合理 2.合约能完整描述业务对象，正确表达业务对象、实体等之间的关系 3.正确编写智能合约设计文档	5
	智能合约开发	使用Solidity语言进行智能合约开发，完成智能合约部署和调用	1.合约编写功能覆盖全面、逻辑正确 2.正确部署和调用合约	20
	智能合约测试	编写智能合约单元测试代码并完成合约功能测试、性能测试	1.对已有合约正确构建单元测试 2.正常使用工具完成合约功能及性能测试	5
模块三 区块链应用 系统开发	区块链应用前端开发	根据业务需求，使用前端开发框架完成页面设计，使用已提供的服务端接口获取业务数据，并进行部署展示	1.正确编写前端代码，完成服务端接口调用 2.正确完成前端数据展示及页面逻辑	10
	区块链应用后端开发	依据功能需求，使用Java等后端开发语言及常用框架进行后端代码开发，访问数据库、实现应用程序接口、调用智能合约，完善区块链应用系统，完成后端代码的部署	1.正确使用后端开发语言和框架，完成数据库调用等功能，实现应用程序接口2.正确编写后端接口程序，调用智能合约，实现链上信息的查询和结果展示 3.正确编写后端接口程序，进行区块链应用操作 4.正确部署后端程序	20
职业素养		文档写作科学规范，具有团队合作精神和创新意识，比赛操作严谨，代码编写规范，文明竞赛	1.分工合理 2.操作规范 3.文明竞赛	5

（三）评分方式

1.参与赛项成绩管理的组织机构包括裁判组、监督仲裁组，裁判组实行“裁判长负责制”，设裁判长 1 名，全面负责赛项的裁判与管理工作，裁判组负责评分、加密和现场执裁。

2.评分裁判负责对参赛队的竞赛成果按赛项评分标准进行评定。

3.赛项采取两次加密原则，加密裁判负责对参赛选手抽签进行加密并且保密，不得将任何信息透露给其他人员，否则按照相关规定予以处理。

4.现场裁判负责对整个赛场进行巡查和监督，必须严格按照现场裁判要求做好相应工作。

5.监督组对裁判组的工作进行全程监督，并对竞赛成绩抽检复核。仲裁组负责接受由参赛队领队提出的书面申诉，组织复议并及时反馈复议结果。

6.赛项成绩解密后，经裁判长、监督组签字后，在大赛办指定的地点，以纸质形式向全体参赛队进行公布。成绩公布 2 小时无异议后，将赛项总成绩的最终结果提交重庆市大赛办。

十二、奖项设置

本赛项设参赛选手团体一、二、三等奖。奖项设定以赛项实际参赛队总数为基数，一、二、三等奖获奖比例分别为 10%、20%、30%（小数点后四舍五入）。获得一等奖的参赛队指导教师获“优秀指导教师奖”。

奖项获得根据参赛队最终成绩由高到低进行排序，如出现参赛队最终成绩并列的情况，按照模块一、二、三顺序的得分高低排序，即总成绩相同的情况下比较模块一的成绩，模块一成绩高的排名优先，如果模块一成绩也相同，则按模块二的成绩进行排名，以此类推完成相同成绩的排序。

十三、赛项预案

（一）竞赛环境突发应急预案

1.赛场备用工位：赛场提供占总参赛队伍10%的备用工位。

2.竞赛系统可靠性：竞赛系统使用的服务器应进行冗余，数据库、存储应使用高可用架构。

3.服务器资源问题：若服务器在比赛过程中出现卡顿、死机等情况，在现场裁判与技术人员确定情况后，可更换服务器资源。

4.PC 机问题：若 PC 机在比赛过程中出现死机、蓝屏等现象（重启后无法解决），在现场裁判与技术人员确定情况后，可启用备用工位或更换PC机。

（二）参赛选手突发情况应急预案

竞赛期间，承办校须安排 2 名医生现场值班，安排好车辆随时待命，一旦参赛选手出现发病、受伤、意外伤害或者食品安全事故，需及时对选手进行现场救治或送医治疗。

（三）停水停电及火情突发应急预案

1.承办校后勤部门须保证竞赛期间供电、供水正常遇停电及时启用自备电源（或租赁发电机）供电，保证竞赛设备用电正常。

2.发现火情时立即根据火情状况，决定是否组织人员疏散，是否切断电源和光源，以及是否需要报警。

（四）治安事件突发应急预案

规范赛场秩序，加强法制和安全教育，对发现有情绪异常、行为过激的选手，及时与参赛队领队沟通联系，做好劝导和化解工作。

十四、竞赛须知

（一）参赛队须知

1.参赛队名称需统一使用规定的学校代表队名称，不使用其他组织、团体的名称，参赛 2 支队队伍的，由参赛校指定参赛队为 XX 学院 1 队、2 队。

2.各参赛院校应指定 1 名负责人任赛项领队，全权负责参赛事务的组织、协调和领导工作。

3.参赛队按照赛项赛程安排，凭参赛证、身份证或护照、学生证参加比赛及相关活动。

4.主办方统一安排各参赛队在比赛前一天进入赛场熟悉环境和设施情况。

5.参赛队选手、领队和指导教师要有良好的职业道德，严格遵守比赛规则和比赛纪律，服从裁判，尊重裁判和赛场工作人员，自觉维护赛场秩序。

6.领队应负责赛事活动期间本队所有选手的人身及财产安全，如发现意外事故，应及时向赛项执委会报告。

7.各学校组织代表队时，应为参赛选手购买大赛期间的人身意外伤害保险，参赛选手在参赛期间的人身安全由参赛院校负责，承办校不承担相关责任。

（二）领队、指导教师须知

1.严格遵守赛场的各项规定，服从裁判，文明竞赛。如发现弄虚作假者，取消参赛资格，名次无效。

2.领队和指导教师务必带好有效身份证件，在活动过程中佩戴“领队证”“指导教师证”参加竞赛相关活动。

3.领队要严格遵守竞赛的各项规定，加强对参赛人员的管理，做好赛前准备工作，督促选手带好证件等竞赛相关材料。

4.在比赛期间要严格遵守比赛规则，不得私自接触裁判人员。

5.竞赛过程中，未经裁判许可，领队、指导教师及其他人员一律不得进入竞赛现场。

6.如对竞赛过程有疑议，由领队负责以书面形式向大赛仲裁组反映，但不得影响竞赛进行。

7.对申诉的仲裁结果，领队要带头服从和执行，并做好选手工作。参赛选手不得因申诉或对处理意见不服而停止竞赛，否则以弃权处理。

8.领队和指导老师应及时查看有关赛项的通知和内容，认真研究和掌握本赛项竞赛的规程、技术规范和赛场要求，指导选手做好赛前的一切技术准备和竞赛准备。

（三）参赛选手须知

1.参赛选手应严格遵守赛场规章、操作规程和工艺准则，保证人身及设备安全，接受裁判员的监督和警示，文明竞赛。

2.选手应按照规定时间抵达赛场，凭统一发放的参赛证、身份证或护照、学生证完成入场检录、抽签确定竞赛工位号，不得迟到早退。

3.参赛选手进入赛场前，须将身份证、学生证交由检录人员统一保管，不得带入场内。参赛证始终佩戴，以备检查。

4.参赛选手凭竞赛工位号进入赛场，不允许携带任何书籍和其他纸质资料，竞赛统一提供草稿纸。不允许携带任何电子设备及通信工具和存储设备（如U盘），竞赛统一提供计算机以及应用软件。

5.参赛选手应在规定的时间段进入赛场，认真核对竞赛工位号，在指定位置就座。入场后，赛场工作人员与参赛选手共同确认操作条件及设备状况，填写相关确认文件，并由参赛队长确认签字（签竞赛工位号）。

6.参赛选手在收到开赛信息前不得启动操作。在竞赛过程中，确因计算机软件或硬件故障，致使操作无法继续的，经裁判长确认，予以启用备用计算机。

7.参赛选手应在竞赛规定的时间完成任务书内容，并按要求提交成果物。

8.参赛选手需及时保存工作记录。对于因自身原因造成的数据丢失，由参赛选手自行负责。

9.参赛队所提交的答卷采用竞赛工位号进行标识，不得出现地名、校名、姓名、参赛证编号等信息，否则取消竞赛成绩。

10.竞赛过程中，因严重操作失误或安全事故不能进行比赛的，现场裁判员有权中止该队比赛。

11.参赛期间，食品、饮水等由赛场统一提供，选手休息和如厕时间均计算在比赛时间内。

12.在参赛期间，选手应注意保持工作环境及设备摆放符合企业生产“5S”（即整理、整顿、清扫、清洁和素养）要求。

13.在比赛中如遇非人为因素造成的设备故障，经裁判确认后，可向裁判长申请补足排除故障的时间。

14.参赛选手原则上不得提前结束比赛。如确因不可抗因素需要提前结束比赛的，须向现场裁判员举手示意，经裁判长许可并完成记录后，方可离开。

15.竞赛时间结束，全体选手应立刻起立，结束操作，将资料和工具整齐摆放在操作平台上，将竞赛成果物按照赛题指定的方式提交，并由现场裁判和参赛队长共同签字确认后方可离开赛场，离开赛场时不得带走任何资料。

16.参赛选手未能按时提交竞赛成果物的，竞赛成绩计为零分。

17.在竞赛期间，未经赛项组委会批准，参赛选手不得接受其他单位和个人进行的与竞赛内容相关的采访。参赛选手不得将竞赛的相关信息私自公布。

18.符合下列情形之一的参赛选手，经裁判组裁定后终止其竞赛：

（1）不服从裁判员管理、扰乱赛场秩序、干扰其他参赛选手比赛，裁判员应对其提出警告。二次警告后无效，或情节特别严重造成竞赛中止的，经裁判长确认，终止其比赛，并取消竞赛资格和竞赛成绩。

（2）竞赛过程中，如选手人为造成计算机、仪器设备及工具等严重损坏，负责赔偿其损失，裁判长有权裁定其结束竞赛、取消竞赛成绩或取消竞赛资格。

（3）竞赛过程中，造成重大安全事故，或产生重大安全事故隐患，经裁判员提示没有采取措施的，裁判员可暂停其竞赛，由裁判长裁定其结束竞赛、取消竞赛成绩或取消竞赛资格。

（四）工作人员须知

1.树立服务观念，注意文明礼貌，全程佩带证件，保持良好形象。

2.认真学习赛项指南，以高度负责的精神、严肃认真的态度和严谨细致的作风，认真履行岗位职责。

3.在赛项执委会的领导下，服从调配和分工，确保比赛工作的顺利进行。

4.自觉遵守赛项纪律和规则，保守秘密。

5.严守工作岗位，按时到岗，不无故离岗，特殊情况需向赛项执委会请假。

6.熟悉应急预案，如遇突发事件，及时组织疏散，确保人员安全。

7.工作人员在比赛中若有舞弊行为，立即撤销其工作资格，并按规定严肃处理。

8.保持沟通，加强协作，提高工作效率。

十五、申诉与仲裁

1.各参赛队对不符合大赛和赛项规程规定的仪器、设备、工装、材料、物件、计算机软硬件、竞赛使用工具、用品，竞赛执裁、赛场管理，以及工作人员的不规范行为等持有异议时，由各参赛队领队向赛项监督仲裁工作组提出书面申诉。

2.监督仲裁人员的姓名、联系方式、工作地点应该在竞赛期间向参赛队和工作人员公示，确保信息畅通并同时接受大众监督。

3.赛项监督仲裁组只接受各学校领队签字、递交的仅限于本队的书面申诉报告。

4.提出申诉的时间应在比赛成绩公示后 2 小时内，超过时效不予受理。申诉报告应对申诉事件的现象、发生时间、涉及人员、申诉依据等进行充分、实事求是的叙述。

5.赛项监督仲裁工作组在接到申诉报告后的 2 小时内组织复议，并及时将复议结果以书面形式告知申诉方。申诉方对复议结果仍有异议，可由参赛队学校向重庆市大赛办提出申诉。大赛办的仲裁结果为最终结果。

6.仲裁结果由申诉人签收，不能代收。如在约定时间和地点申诉人离开，视为自行放弃申诉。

7.申诉方可随时提出放弃申诉。

8.申诉方必须提供真实的申诉信息并严格遵守申诉程序，提出无理申诉或采取过激行为扰乱赛场秩序的，应视情节给予取消参赛成绩等处罚。

附件1

2023-2024年重庆市职业院校技能大

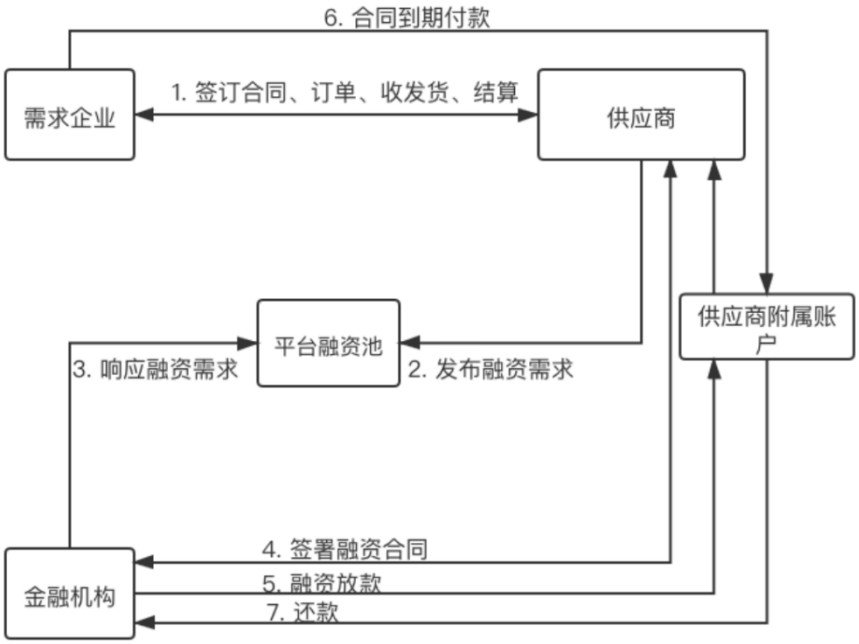
赛区块链技术应用赛项

样卷

背景描述

在供应链金融这个万亿级市场中，区块链正在快速商业化落地，助力产业革新。基于区块链的供应链金融业务的理念是：以源自企业的应收账款为底层资产，通过区块链技术实现债券凭证的转让拆分。其中，在原始资产上链时，通过对应收账款进行审核校验，确认贸易关系和身份真实有效，和保证上链资产的真实可信。再者，债权凭证可基于供应链进行层层拆分与流转，都可完整追溯到最底层资产，以实现核心企业和金融机构对供应商的“信用穿透”。

某公司规划开发一个区块链供应链金融平台，包括核心企业、供应商、银行等角色，通过智能合约代码逐步构建区块链供应链金融平台的基本功能，实现银行向核心企业提供授信并发行数字凭证，企业与企业之间转让数字凭证。此外需要完成区块链供应链金融平台的前后端，实现基本的业务逻辑。



模块一：区块链产品方案设计及系统运维（35分）

任务1-1：区块链系统部署与运维（25分）

围绕供应链金融区块链平台部署与运维需求，进行项目相关系统、节点以及管理工具的部署工作。通过监控工具完成对网络、节点服务的监控。最终根据业务需求规范，完成系统日志、网络参数、节点服务等系统结构的维护。

1. 根据参数与端口设置要求，部署区块链系统并验证；
2. 根据参数与端口设置要求，部署区块链网络管理平台并验证；
3. 基于区块链系统相关管理平台，按照任务指南实施系统运维工作并验证。
4. 基于区块链系统相关监管工具，按照任务指南对区块链系统进行监管。

子任务 1-1-1： 搭建区块链系统并验证（8 分）

基于给定服务器环境以及软件（地址“/root/tools”），使用 Docker 以默认配置安装单机 4 节点的区块链系统，并完成控制台工具的部署：

(1) 完成系统搭建配置与启动。（2 分）

(2) 使用基于 Docker 命令查看区块链系统状态。（2 分）

(3) 检查区块链系统节点 node0 连接状态输出。（2 分）

(4) 配置控制台，管理相关证书并启动。（2 分）

子任务 1-1-2：区块链管理平台部署与验证（8 分）

基于给定服务器环境以及软件（地址“/root/tools”），按要求部署区块链管理平台，具体工作如下：

(1) 配置 Mysql 数据库（2 分）

(2) 配置管理平台连接区块链系统（2 分）

(3) 使用命令启动管理平台服务（2 分）

(4) 验证管理平台启动情况（2 分）

子任务1-1-3：区块链系统节点运维（5分）

基于已完成的区块链系统与管理平台搭建工作，开展相关节点运维工作：

(1) 生成新节点(node4)，启动并检查（2 分）

(2) 修改新节点配置，并查看节点的 nodeid（2 分）

(3) 将新节点作为观察节点加入 group1 当中，并检查是否加入成功（1 分）

子任务1-1-4：区块链系统管理平台运维（4分）

基于已部署的区块链系统管理平台，进行系统相关运维工作：

(1) 基于管理平台功能页面，添加新主机（2 分）

(2) 基于管理平台功能页面，修改新节点（node4）节点状态，并监控。（2 分）

任务1-2：区块链系统测试（10分）

设计对区块链系统的测试流程；结合实际业务需求，调用部署的智能合约中进行系统测试、性能测试等；根据业务需求，分析并且修复给定智能合约中的安全漏洞。利用模拟业务和测试工具来完成对区块链系统服务数据的测试。

1.基于区块链系统的中间件服务的部署脚本完成中间件服务环境搭建以及搭建结果验证，最后将执行结果截图保存。（3 分）

(1) 实现区块链系统中间件服务平台部署。（1 分）

(2) 实现区块链系统中间件服务签名功能启动情况验证。（1 分）

(3) 区块链中间件服务节点管理进程启动情况验证和浏览器验证。（1 分）

2.智能合约安全漏洞测试。（7 分）

有如下智能合约：

```
pragma solidity ^0.7.6;

contract TimeLock {
    mapping(address => uint) public balances;
    mapping(address => uint) public lockTime;

    function deposit() external payable {
        balances[msg.sender] += msg.value;
        lockTime[msg.sender] = block.timestamp + 1 weeks;
    }

    function increaseLockTime(uint _secondsToIncrease) public {
        lockTime[msg.sender] += _secondsToIncrease;
    }

    function withdraw() public {
        require(balances[msg.sender] > 0, "Insufficient funds");
        require(block.timestamp > lockTime[msg.sender], "Lock time not expired");

        uint amount = balances[msg.sender];
        balances[msg.sender] = 0;

        (bool sent, ) = msg.sender.call{value: amount}("");
        require(sent, "Failed to send Ether");
    }
}

contract Attack {
    TimeLock timeLock;

    constructor(TimeLock _timeLock) {
        timeLock = TimeLock(_timeLock);
    }

    fallback() external payable {}
```

```
    function attack() public payable {
        timeLock.deposit{value: msg.value}();
        timeLock.increaseLockTime(
            type(uint).max + 1 - timeLock.lockTime(address(this))
        );
        timeLock.withdraw();
    }
}
```

如上代码主要实现功能为规定了转账冻结时间，在冻结时间内用户不能提取存款的金额。

（1）分析智能合约中存在问题，并说明危害。（2 分）

（2）根据测试工具中的代码文件，编写测试用例，复现智能合约中存在的漏洞。（3分）

（3）创建新的智能合约，修复其中问题，说明修复内容并测试。（2 分）

模块二：智能合约开发与测试（30分）

任务2-1：智能合约开发（20分）

使用Solidity语言完成智能合约开发、部署和调用，要求如下：

1.供应链金融实体信息编码（6分）

（1）编写供应链金融智能合约的实体接口，完成实体通用数据的初始化，实现企业和票据实体信息上链的功能；（2分）

表2.2.2.1 SupplyChain实体说明

名称	类型	说明
companyName	string	公司名称
companyAddress	address	公司地址
creditAsset	uint	信用资产
acceptReceiptIndex	uint[]	接收的凭证
sendReceiptIndex	uint[]	发送的凭证
senderAddress	address	发送票据的地址
accepterAddress	address	接收票据的地址
receiptType	uint8	凭证类型
transferType	uint8	交易类型
amount	uint	交易数量

```
//公司信息结构体
    struct Company {
        //①公司名称
        //②公司地址
        //③信用资产
        //④接收的凭证
        //⑤发送的凭证
    }

//数字发票收据信息
    struct Receipt {
        //⑥发送票据的地址
```

```
        //⑦接收票据的地址
        //⑧凭证类型
        //⑨交易类型
        //⑩交易数量
    }
```

（2）编写企业上链信息接口，实现供应链金融的企业信息上链；（2分）

```
function addCompany(string name, address companyAddress) returns(bool) {
        //①实例化公司
        //②添加公司地址
        //③将实例化的公司添加到公司映射
        //④返回添加成功标识
    }
```

（3）基于给定的智能合约代码以及注释，完成银行向企业交易的接口函数；（2分）

```
function bankToCompanyReceipt(address senderAddress, address accepterAddress,
uint amount, uint8 receiptType) returns(uint) {
①判断接收地址存在
        ②实例化银行
        ③实例化公司

        if (keccak256(bank.bankName) == keccak256("")) {
            return 404001;
        }

        //确认公司存在
        if (keccak256(company.companyName) == ④) {
            return 404002;
        }

        if (bank.creditAsset < amount) {
            return 500001;
        }
```

2.供应链金融公司与公司接口编码（6分）

（1）编写公司与公司之间进行交易的历史存证上链接口，实现公司与公司之间的交易功能；（2分）

```
    function companyToCompanyReceipt(①, address accepterAddress, uint amount,
uint8 receiptType) returns(uint) {

            //②接收地址判断
            Company memory senderCompany = companyMap[③];
            Company memory ④ = companyMap[accepterAddress];
            //确认发送公司存在
            if (keccak256(senderCompany.⑤) == keccak256("")) {
                return 404001;
            }

            //确认接收公司存在
            if (keccak256(accepterCompany.companyName) == ⑥) {
                return 404002;
            }

            //如果存证接收的公司资产小于存证数额，那么就不能交易发送存证
            if (accepterCompany.creditAsset ⑦ ⑧) {
                return 500001;
            }
```

（2）编写创建存证的接口，实现创建存证的功能；（2分）

```
    Receipt memory newReceipt = Receipt(①, accepterAddress, receiptType, 2,
amount);
            receiptIndex += 1;
            //记录存证（存证Map，公司Map对应地址的发送和接收存证列表）
            receiptMap[receiptIndex] = ②;
            companyMap[③].sendReceiptIndex.push(receiptIndex);
            companyMap[accepterAddress].acceptReceiptIndex.push(④);
```

（3）编写交易金额数量变化的接口，实现凭证交易双方资金的变化功能；（2分）

```
    companyMap[①].creditAsset ② amount;
            companyMap[③].creditAsset ④ amount;
            return 200;
        }
```

3.供应链金融公司与银行交易的接口编码（4分）

（1）编写公司与银行之间进行交易的历史存证上链接口，实现公司与银行之间的交易功能；（2分）

```
function companyToBankReceipt(address senderAddress, ①, uint amount, uint8 receiptType) returns(uint) {

        ②

        Bank memory bank = bankMap[senderAddress];
        Company memory accepterCompany = companyMap[③];

        //确认发送公司存在
        if (keccak256(bank.bankName) == ④) {
            return 404001;
        }

        //确认接收公司存在
        if (keccak256(accepterCompany.companyName) == keccak256("")) {
            return 404002;
        }

        //如果存证接收的公司资产小于存证数额，那么就不能交易发送存证
        if (accepterCompany.creditAsset < amount) {
            return 500001;
        }
```

（2）编写创建存证的接口，实现创建存证的功能；（1分）

```
//创建存证
        Receipt memory newReceipt = Receipt(senderAddress, accepterAddress, ①, 3, amount);
        receiptIndex ② 1;
        receiptMap[③] = newReceipt;
        bankMap[senderAddress].sendReceiptIndex.push(receiptIndex);
        companyMap[accepterAddress].④;
```

（3）编写交易金额数量变化的接口，实现凭证交易双方资金的变化功能；（1分）

```
bankMap[senderAddress].① ② amount;
companyMap[accepterAddress].③ ④ amount;
        return 200;
}
```

4. 合约编译、部署和调用（4 分）

（1）解决代码错误和警告，正确编译并部署合约，成功获取部署的合约地址和 abi。（1 分）

（2）调用食品溯源智能合约的接口，完整验证业务流程。（3 分）

任务 2-2：智能合约测试（10 分）

编写智能合约单元测试代码并完成合约功能测试、性能测试，具体要求如下：

1.配置区块链网络（2分）

启动区块链网络，创建新的Workspace，配置对外访问的RPC接口为7545，配置项目的配置文件config.js实现与新建Workspace的连接。

2.补充给定基础代码中注释提示的部署逻辑（2分）

基于VSCODE加载的测试项目，补全位于test文件夹中HelloWorld.js文件预操作的方法。在测试文件中添加预定义的方法（在其他方法启动前使用）。

3.补充代码中注释提示的测试逻辑（2分）

基于VSCODE加载的测试项目，补全位于test文件夹中HelloWorld.js文件，添加测试用例，测试智能合约的get方法。

4.测试hello.get()方法（2分）

基于VSCODE加载的测试项目，补全位于test文件夹中HelloWorld.js文件，添加测试用例，测试智能合约的hello.get()方法。

5.测试.should.equal进行对比判断（2分）

基于VSCODE加载的测试项目，补全位于test文件夹中HelloWorld.js文件，添加测试用例，测试智能合约的equal字符串比较方法。

模块三：区块链应用系统开发（30分）

任务3-1：区块链应用前端功能开发（10分）

1.请基于前端系统的开发模板，在注册组件Register.vue文件中添加对应的注册逻辑代码，实现对后端系统的注册功能，并测试功能完整性（3分）：

本题目的具体要求如下：

(1) 界面有明确的注册相关提示语

(2) 需要填写的部分有组织名称、区块链地址、组织类型

(3) 页面需要有“返回”按钮，可以跳转到登录页面

(4) 点击“注册”按钮时需要检查区块链地址是否已输入

(5) 注册成功后跳转登录页面

Register.vue:

代码片段1：

```
<el-row>
        <el-col :span="16" :offset="4">
          <el-form label-width="100px">
            <h3>选手填写部分</h3>
            <el-form-item label="组织名称:">
              <el-input type="primary" v-model="选手填写部分"></el-input>
            </el-form-item>
            <el-form-item label="区块链地址:">
              <el-input type="primary" v-model="选手填写部分"></el-input>
            </el-form-item>
            <el-form-item label="组织类型:">
              <el-radio-group v-model="orgType">
              <el-radio :label="1">公司</el-radio>
              <el-radio :label="2">银行</el-radio>
              </el-radio-group>
            </el-form-item>
          </el-form>
        </el-col>
      </el-row>
```

```
      <el-row style="padding-bottom:20px">
        <el-button type="primary" 选手填写部分>注册</el-button>
        <el-button type="primary" 选手填写部分>返回</el-button>
      </el-row>
```

代码片段 2：

```
register: function () {
      if (this.address == "") {
        alert(选手填写部分)
      }else {
        let postData = {
          orgType: 选手填写部分,
          username: 选手填写部分,
          address:选手填写部分
        }
// 和后端交互
        选手填写部分
      }
    },
```

代码片段 3：

```
goback: function () {
      this.orgType = ''
      this.username =  ''
      this.address = ''
      选手填写部分
    }
```

2.请基于前端系统的开发模板，在登录组件 Login.vue 文件中添加对应的登录逻辑代码，实现对后端系统的登录功能，并测试功能完整性（3 分）：

本题目的具体要求如下：

(1) 界面有明确的登录相关提示语

(2) 需要填写的项有用户地址、组织类型

(3) 页面需要有“注册”按钮，可以跳转注册页面

(4) 点击“登录”按钮时需要检查各个表项是否已输入

(5) 登录成功后跳转首页，路径为“/home”

Login.vue:

代码片段 1：

```
<el-col :span="16" :offset="4">
          <el-form label-width="80px">
            <h1>供应链金融应用</h1>
            <h3>选手填写部分</h3>
            <el-form-item label="用户地址:">
              <el-input type="primary" v-model="选手填写部分"></el-input>
            </el-form-item>
            <el-form-item label="组织类型:">
              <el-radio-group v-model="选手填写部分">
              <el-radio :label="1">公司</el-radio>
              <el-radio :label="2">银行</el-radio>
              </el-radio-group>
            </el-form-item>
          </el-form>
        </el-col>


      </el-row>
      <el-row style="margin-bottom: 20px">
        <el-button type="primary" 选手填写部分>登录</el-button>
        <el-button type="primary" 选手填写部分>注册</el-button>
      </el-row>      </el-row>
```

代码片段 2：

```
login: function () {
      if (this.address == "") {
        alert("选手填写部分")
      }else if (this.orgType == "") {
        alert("选手填写部分")
      }else {
        let postData = {
          orgType: 选手填写部分,
          address: 选手填写部分
        }
```

```
        // 与后端交互
选手填写部分
      }
},
```

代码片段 3：

```
register: function () {
         选手填写部分
      },
```

3.请基于前端系统的开发模板，在公司组件 Company.vue 文件中添加对应的逻辑代码，实现对后端系统的公司相关业务功能，并测试功能完整性（2 分）：

Company.vue:

代码片段 1：

```
<el-row>
            <el-col :span="20" :offset="2">
            <el-table :data="companyList" style="font-size: 20px">
              <el-table-column prop="address" label="账户地址"></el-table-
column>
              <el-table-column prop="name" label="公司名称"></el-table-
column>
              <el-table-column prop="amount" label="账户总额"></el-table-
column>
              <el-table-column label="查看详情">
                <template slot-scope="scope">
                  <el-button type="primary" @click="选手填写部分">查询</el-
button>
                </template>
              </el-table-column>
              <el-table-column prop="receiptType" label="转账">
                <template slot-scope="scope">
                  <el-button type="primary" @click="transfer(scope.row)">操
作</el-button>
                </template>
              </el-table-column>
```

```
            </el-table>
          </el-col>
          </el-row>
          <el-row>
```

代码片段 2：

```
<el-dialog title="公司详情" :visible.sync="dialogVisible">
        <el-form label-width="100px">
          <el-form-item label="账户地址:">
            {{选手填写部分 }}
          </el-form-item>
          <el-form-item label="公司名称:">
            {{ 选手填写部分 }}
          </el-form-item>
          <el-form-item label="账户总额:">
            {{ 选手填写部分 }}
          </el-form-item>
        </el-form>
```

代码片段 3：

```
detail: function (queryAddress) {
      this.dialogVisible = true
      let address = 选手填写部分
      this.axios.get(`选手填写部分
?address=${address}&queryAddress=${queryAddress}`)
          .then((response) => {
            console.log(response)
            if (response.data.code == 200) {
              let inAddress = response.data.data.companyVO.address;
              let inName = 选手填写部分;
              let inAmount = response.data.data.companyVO.amount;
              this.companyDetail = {
                address: 选手填写部分,
                name: 选手填写部分,
                amount: 选手填写部分,
```

```
                senderReceiptList: response.data.data.senderReceiptList,
                accepterReceiptList: response.data.data.accepterReceiptList
              }
            } else {
              alert(`请求内容有误，${response.data.data}`)
            }
          })
      },
```

4.请基于前端系统的开发模板，在银行组件 Bank.vue 文件中添加对应的逻辑代码，实现对后端系统的银行相关业务功能，并测试功能完整性（2 分）：

Bank.vue:

代码片段 1:

```
<el-row>
      <el-dialog title="交易（发送凭证）页"
:visible.sync="transDialogVisible" width="30%">
        <el-form label-width="100px">
          <el-form-item label="发送账户地址:">
            {{选手填写部分}}
          </el-form-item>
          <el-form-item label="接收账户地址:">
            {{选手填写部分}}
          </el-form-item>
          <el-form-item label="交易额:">
            <el-col :span="16" :offset="4">
              <el-input type="primary" v-model="选手填写部分"></el-input>
            </el-col>
          </el-form-item>
          <el-form-item label="凭证类型:">
            <el-select v-model="选手填写部分" placeholder="请选择">
              <el-option
                  v-for="item in options"
                  :key="item.value"
```

```
                    :label="item.label"
                    :value="item.value">
                  </el-option>
                </el-select>
              </el-form-item>
            </el-form>
            <el-row>
              <el-button type="primary" size="medium" @click="选手填写部分">确定
</el-button>
            </el-row>
          </el-dialog>
        </el-row>
```

代码片段 2:

```
executeTransaction: function() {
      let funcName = "companyToBankReceipt";
      if (this.transDetail.amount ==选手填写部分) {
        alert('交易额不能为空！')
        return
      }
      if (this.$cookies.get('orgType') == 选手填写部分) {
        alert('银行不能给银行发送凭证！')
        return
      }
      if (选手填写部分) {
        alert("凭证发送账户和接收账户不能相一致！")
        return
      }
      this.axios.post(`/finance/transaction/${funcName}`, 选手填写部分
).then((response) => {
        if (response.data.code == 200) {
          alert('凭证发送成功')
          this.query()
          this.transDialogVisible = false
        }else {
```

```
            alert(`凭证发送失败, ${response.data.data}`)
          }
        })
      },
```

任务3-2：区块链应用后端功能开发（20分）

1.开发区块链供应链金融应用后端系统中用户功能模块对应的用户注册功能，根据前后代码补充最合适的代码，并测试功能完整性。（4 分）

OrgServiceImpl.java：

```
 /**
     * 注册Service
     * RegisterBO registerBO
     * */
    @Override
    public Result<String> register(RegisterBO registerBO) {
        if (StrUtil.isEmpty(选手填写部分) ||
                StrUtil.isEmpty(选手填写部分) ||
                registerBO.getOrgType() == 选手填写部分
        ) {
            return Result.error(ResultVO.PARAM_EMPTY);
        }

        List funcParam = new ArrayList();
        funcParam.add(选手填写部分);
        funcParam.add(选手填写部分);
        if(registerBO.getOrgType() == 2){
            funcParam.add(BigInteger.valueOf(1000));
        }

        String funcName;
        if(registerBO.getOrgType() == 2){
            funcName =选手填写部分;
        }else{
```

```
            funcName =选手填写部分;
        }

        String _result =
weBASEUtils.funcPost(OWNER_ADDRESS,funcName,funcParam);
        JSONObject _resultJson = JSONUtil.parseObj(_result);
        if (_resultJson.containsKey("statusOK") == false ||
_resultJson.getBool("statusOK") != true){ // _resultJson.getInt("code") > 0
            return Result.error(ResultVO.选手填写部分);
        }

        return Result.success("ok");
    }
```

2.开发区块链供应链金融应用中后端系统中用户功能模块对应的用户登录功能，根据前后代码补充最合适的代码，并测试功能完整性。（4 分）

OrgServiceImpl.java:

```
/**
     * 登录Service
     * LoginBO loginBO
     **/
    @Override
    public Result<String> login(@RequestBody LoginBO loginBO) {
        if (StrUtil.isEmpty(loginBO.getAddress())
        ) {
            return Result.error(ResultVO.PARAM_EMPTY);
        }

        List funcParam = new ArrayList();
        funcParam.add(选手填写部分);

        String funcName;
        if(loginBO.getOrgType() == 2){
```

```
            funcName =选手填写部分;
        }else{
            funcName =选手填写部分;
        }

        String _result = weBASEUtils.funcPost(选手填写部分
,funcName,funcParam);
        JSONArray _resultJson = JSONUtil.parseArray(_result);
        if (StrUtil.isEmpty(_resultJson.get(0).toString())){
            return Result.error(ResultVO.选手填写部分);
        }

        return Result.success("ok");
    }
```

3.开发区块链供应链金融应用后端系统中查询功能模块对应的查询所有公司信息功能，根据前后代码补充最合适的代码，并测试功能完整性。（4 分）

QueryServiceImpl.java:

```
/**
     * 获取所有公司数据，不包含存证详细信息
     * */
    @Override
    public Result listCompany(String userAddress){

        String _result =
weBASEUtils.funcPost(userAddress,"getAllCompanyAddress",new ArrayList());

        try {
            JSONArray respArray = 选手填写部分;
            String a = respArray.get(0).toString();
            JSONArray addressArray = 选手填写部分;
            List<CompanyVO> companyList = new ArrayList<>();
            for(Object obj: addressArray) {
                CompanyVO companyVO = 选手填写部分;
```

```
                companyList.add(companyVO);
            }
            return Result.success(选手填写部分);
        }catch (Exception e) {
            ResultVO resultVO = ResultVO.CONTRACT_ERROR;
            resultVO.setData(选手填写部分);
            return Result.error(选手填写部分);
        }
    }
```

4.开发区块链供应链金融应用的后端系统中查询功能模块对应的查询银行凭证列表功能根据前后代码补充最合适的代码，并测试功能完整性。（4 分）

QueryServiceImpl.java:

```
/**
     * 获取包括银行详情，银行发送凭证列表，银行接收凭证列表
     * */
    @Override
    public Result getBankEntity(String userAddress, String queryAddress){
        BankVO bankVO = 选手填写部分;
        List<ReceiptVO> senderReceiptList = 选手填写部分;
        List<ReceiptVO> accepterReceiptList = 选手填写部分;
        BankEntityVO bankEntityVO = new BankEntityVO();
        bankEntityVO.setBankVO(选手填写部分);
        bankEntityVO.setSenderReceiptList(选手填写部分);
        bankEntityVO.setAccepterReceiptList(选手填写部分);
        return Result.success(bankEntityVO);
    }
```

5.开发区块链供应链金融应用的后端系统中交易功能模块对应的银行向公司交易凭证功能，根据前后代码补充最合适的代码，并测试功能完整性。（2 分）

TransactionServiceImpl.java:

```
/**
     * 公司向银行发送凭证（银行转账给公司）
     * */
```

```
        public Result<String> bankToCompanyReceipt(TransactionBO transactionBO)
{
            选手填写部分;
    }
```

6.开发区块链供应链金融应用的后端系统中交易功能模块对应的公司向公司交易凭证功能，根据前后代码补充最合适的代码，并测试功能完整性。（2 分）

TransactionServiceImpl.java：

```
 /**
     * 公司向公司的凭证发送
     * */
        public Result<String> companyToCompanyReceipt(TransactionBO
transactionBO) {
            选手填写部分;
        }
```