LabVIEW高级架构师认证 CLA的考试范围：

认证ID	认证名称	工作描述
CLA	LabVIEW高级架构师认证	根据给定的一个大型应用程序的一系列要求，LabVIEW高级架构师能够开发、领导和指导一个开发团队，创建一个有效的、高性价比的方案。

任务ID	任务	考察目标ID	考察目标
CLA-ADT-01	在LabVIEW应用程序 中，运用标准的、普遍接受的高级设计技术	CLA-ADT-01-01	为开发大型LabVIEW应用程序制定优先级计划： ● 恰当的客户/设计者交互 ● 识别后续项、考虑因素和突出的问题 ● 生成该应用程序的计划和估算
		CLA-ADT-01-02	为多人开发的LabVIEW应用程序创建设计原则： ● 代码段间的交互 ● 数据需求
		CLA-ADT-01-03	给定一组条件，以通用的书面格式创建一个应用程序的用例（不要求使用UML或伪码）
		CLA-ADT-01-04	为一个应用程序创建一组清晰的、精简的、可测试的软件需求
		CLA-ADT-01-05	创建测试计划以实现测试需求
		CLA-ADT-01-06	描述在创建可扩展的、可维护的应用程序中使用编程架构的好处
CLA-ADT-02	在团队中采用合适的编程风格，生成一个LabVIEW应用程序	CLA-ADT-02-01	描述与个人和团队开发有关的恰当的程序风格、代码标准和问题，包括： ● VI命名规范 ● 源码控制 ● 子VI集成 ● 一致的用户界面 ● 可扩展性 ● 文档管理 ● 固定编码的路径、名称或相关信息 ● 变量的过多使用 ● 过于庞大的图表 ● 使用合适的数据类型 ● 标记从移位寄存器引出的连线 ● 顺序结构和顺序局部变量的恰当使用

		CLA-ADT-02-02	生成在团队环境中可以使用的模板，以鼓励适当的编程风格
CLA-ADT-03	执行LabVIEW应用程序的代码检查	CLA-ADT-03-01	描述恰当的文档化在团队开发和代码维护中的重要性
		CLA-ADT-03-02	给定一组条件，检查LabVIEW应用程序的整体设计，包括： ● 文档管理 ● 标记的结构 ● VI功能的描述 ● 足够详细的细节 ● 自说明的代码 ● 效率 ● 编程风格 ● 可读性 ● 合适的前面板设计
		CLA-ADT-03-03	给定一组条件，评估LabVIEW应用程序的编程考虑因素，包括： ● 算法的有效性 ● 数据流编程技术的正确使用 ● 有效的数据/内存管理 ● 常见的编码缺陷包括： ✓ 待确定的引用 ✓ 无效引用 ✓ 潜在的竞争状态
		CLD-VPP-03-04	检查LabVIEW应用程序的运行时行为和内存管理问题，包括： ● 定位算法 ● 使用Refnums还是数值 ● 使用属性节点还是数值 ● 连线分支 ● 执行与操作数据 ● 数据类型的存储 ● 屏幕刷新 ✓ 大量内存的转移 ✓ 同步、异步刷新 ✓ 缓存转移 ✓ 执行顺序 ✓ 分支连线上的破坏性与非破坏性缓存读取
CLA-ADT-04	使用源码控制	CLA-ADT-04-01	描述在多人开发环境中源码控制的必要性

		CLA-ADT-04-02	讨论源码控制与项目管理间的联系
		CLA-ADT-04-03	描述在项目中实施源码控制可能遇到的陷阱
		CLA-ADT-04-04	描述VI交叉连接及其对源码控制的影响
		CLA-ADT-04-05	阐明源码控制对应用程序发布的影响
CLA-ADT-05	设计和开发面向对象的代码	CLA-ADT-05-01	描述LabVIEW应用程序在哪些条件下，适合运用面向对象的编码技术
		CLA-ADT-05-02	阐明创建面向对象应用程序时所必需的设计和实现方面的考虑因素
		CLA-ADT-05-03	在应用程序中恰当地采用功能全局变量
		CLA-ADT-05-04	描述信号量和同步技术的使用
		CLA-ADT-05-05	描述数据和功能的封装
		CLA-ADT-05-06	阐明用于开发可扩展应用程序的方法
		CLA-ADT-05-07	创建VI类和类型定义 用于识别类对象的单个枚举 Datalog 引用句柄
CLA-ADT-06	使用插件技术开发LabVIEW应用程序	CLA-ADT-06-01	阐明通过引用节点使用VI服务器调用的方法
		CLA-ADT-06-02	描述使用VI服务器调用VI的方法，如通过： ● 运行方法 ● 引用节点的调用
		CLA-ADT-06-03	讨论在LabVIEW应用程序中使用插件架构的优缺点
		CLA-ADT-06-04	比较在创建应用程序时使用运行方法和通过引用调用的优缺点
		CLA-ADT-06-05	阐明采用运行方法的VI服务器设置输入数值的过程
		CLA-ADT-06-06	生成调用插入式VI的标准VI
		CLA-ADT-06-07	设计VI接口，以正确识别可作为匹配插件的VI
CLA-ADT-07	使用消息传递架构开发LabVIEW应用程序	CLA-ADT-07-01	描述创建与顶层应用VI分离的用户界面的理由
		CLA-ADT-07-02	创建一个具有与顶层VI分离的用户界面的LabVIEW应用程序 ● 动态调用该用户界面 ● 通过公共消息传递架构与该用户界面通信
		CLA-ADT-07-03	阐明控件引用句柄和控件属性节点的使用和恰当应用
		CLA-ADT-07-04	创建一个使用控件引用句柄和控件属性节

			点以更新其它VI中的控件的应用程序
		CLA-ADT-07-05	创建一个使用消息传递架构在并行循环间传递数据的应用程序
		CLA-ADT-07-06	开发可重入的VI，并描述可重入性对调试工具的影响
		CLA-ADT-07-07	阐明VI间通信时常见方法的使用： ● 队列 ● 通知器 ● VI服务器 ● TCP/IP
		CLA-ADT-07-08	生成一个并行执行多个代码段并以下列方式通信的应用程序： ● 队列 ● 通知器 ● VI服务器 ● TCP/IP
CLA-ADT-08	在LabVIEW中创建或采用模块化代码	CLA-ADT-08-01	描述后向兼容模块化代码的重要性
		CLA-ADT-08-02	阐明一个系统中使用多个同名VI的含意和风险
		CLA-ADT-08-03	描述使用项目专用的全局VI的含意
		CLA-ADT-08-04	生成采用一般命名传统的VI
		CLA-ADT-08-05	在保持后向兼容性的同时升级代码模块
		CLA-ADT-08-06	将子VI归类，以得到有效的模块化组织
CLA-ADT-09	布置一个LabVIEW应用程序	CLA-ADT-09-01	描述在开发大型应用程序时必须考虑的、与布置相关的设计因素
		CLA-ADT-09-02	给定一组条件，详细说明必须在目标机器布置的项
CLA-ADT-10	优化LabVIEW应用程序中执行系统和优先级的使用	CLA-ADT-10-01	阐明下列各项的正确使用和误用带来的陷阱： ● 执行系统 ● 优先级设置 ● 子程序优先级
		CLA-ADT-10-02	优化LabVIEW应用程序的性能
CLA-ADT-11	在LabVIEW应用程序中实现恰当的错误处理技巧	CLA-ADT-11-01	描述错误处理器的重要属性
		CLA-ADT-11-02	阐明应当处理错误和报告错误的情况
		CLA-ADT-11-03	为大型LabVIEW应用程序生成一个可执行的错误处理排列： ● 生成恰当的额外信息并将其附加到错误处理系统 ● 恰当地过滤错误 ● 恰当地处理或传递错误

			如果设置了相应条件（通常设置调试标记），以日志的方式记录错误
CLA-ADT-12	在LabVIEW应用程序 中生成递归代码	CLA-ADT-12-01	在递归方法可取时，描述递归及其环境
		CLA-ADT-12-02	阐明在递归调用时VI服务器的使用
			描述在LabVIEW中创建递归程序的两个常用方法，以及两种方法的优缺点： ● 迭代-递归算法 递归VI
		CLA-ADT-12-03	给定一组需求，在LabVIEW中生成一个递归程序