Certification Overview

The National Instruments LabVIEW Certification Program consists of the following three certification levels:

- Certified LabVIEW Associate Developer (CLAD)
- Certified LabVIEW Developer (CLD)
- Certified LabVIEW Architect (CLA)

Each level is a prerequisite for the next level of certification.

A CLAD demonstrates a broad and complete understanding of the core features and functionality available in the LabVIEW Full Development System and possesses the ability to apply that knowledge to develop, debug, and maintain small LabVIEW modules. The typical experience level of a CLAD is approximately 6 to 9 months in the use of the LabVIEW Full Development System.

A CLD demonstrates experience in developing, debugging, and deploying and maintaining medium to large scale LabVIEW applications. A CLD is a professional with an approximate cumulative experience of 12 to 18 months developing medium to large applications in LabVIEW.

A CLA demonstrates mastery in architecting LabVIEW applications for a multi-developer environment. A CLA not only possesses the technical expertise and software development experience to break a project specification into manageable LabVIEW components but has the experience to see the project through by effectively utilizing project and configuration management tools. A CLA is a professional with an approximate cumulative experience of 24 months in developing medium to large applications in LabVIEW.

Note The CLAD certification is a prerequisite to taking the CLD exam.
The CLD certification is a prerequisite to taking the CLA exam.
There are no exceptions to this requirement for each exam.

Exam Overview

Product: LabVIEW Full Development System version 2012 for Windows. Refer to LabVIEW Development Systems comparison for details on the features available in the LabVIEW Full Development System.

Exam Duration: 1 hour
Number of Questions: 40
Style of Questions: Multiple-choice
Passing grade: 70%

The exam validates application knowledge and not the ability to recall menu steps or names of VIs and components.

The use of LabVIEW or any other external resources is prohibited during the exam. For assistance and wherever appropriate, screenshots from the *LabVIEW Help* are provided in the exam.

To maintain the integrity of the exam, you may not copy or reproduce any section of the exam. Failure to comply will result in failure. In areas where the exam is deployed as a paper based exam, detaching the binding staple will result in failure without evaluation.

Exam Topics

1. LabVIEW Fundamentals
2. Debugging
3. Structures
4. Programming VIs and functions
5. Data communication and synchronization
6. Error Handling
7. Design Patterns
8. VI Server

Exam Topics (Overview):

Topic	SubTopic
1. LabVIEW Fundamentals	a) LabVIEW Programming Principles
	b) LabVIEW Environment
	c) Data Types
2. Debugging	a) Debugging tools
	b) Debugging practices and techniques for different situations
3. Structures	a) Loops
	b) Case Structures
	c) Sequence Structures
	d) Event Structures
	e) Other Structures
4. Programming VIs and functions	a) General Programming Functions
5. Data communication and synchronization	a) Data communication and synchronization
6. Error Handling	a) Error clusters and wires
	b) Error and Dialog VIs
	c) Custom error codes
7. Design Patterns	a) Design Patterns
	b) Sub VI Creation
8. VI Server	a) Configuration
	b) References, Property Nodes, Invoke Nodes

CLAD Topics Details

1. **LabVIEW Fundamentals**
   a. LabVIEW Programming Principles
      1. Objects from the front panel and block diagram
         a) Select the most appropriate object for the front panel window of an application
         b) Describe the connection between an object on the front panel window and its terminal
         c) Select between an object on the front panel window or a block diagram constant
         d) Determine the most appropriate mechanical action for Boolean controls
         e) Determine the appropriateness of Property Nodes and select the appropriate property
   b. LabVIEW Environment
      1. Front panel window, block diagram, and connector pane
         a) Identify the relationship between front panel window and block diagram objects in a VI and their connections through the connector pane
         b) Identify which types of VIs do not have a block diagram
         c) Identify the purpose of the connector pane and icon
         d) Identify the type of palettes and their functions
      2. Identify the configuration options for the following
         a) Front panel window
         b) Block diagram
         c) Environment
   c. Data Types
      1. Data Types and data structures
         a) Select the most appropriate data type for front panel window and block diagram objects
         b) Identify data representation range limitations and wrap-around with different types of integers
         c) Select the most appropriate method to group related data items
         d) Describe the waveform data type and use it to display data on graphs and charts
         e) Describe the timestamp data type and use it to time stamp measurement data
      2. Working with front panel window objects and data types
         a) Determine the most appropriate representation, range, format, precision, and scaling to represent a data item
         b) Identify and describe the scenarios in which you would need to customize a control
         c) Distinguish between a type definition and a strict type definition
         d) Identify and describe the applications which would benefit from the use of a type definition or a strict type definitions

2. **Debugging: Debugging tools and techniques**
   a. Debugging tools
      1. Identify and explain the implications of the VI Properties settings that determine how LabVIEW handles errors and warnings
      2. Identify the errors in a VI that result in a broken **Run** button and use the **Error list** window to determine the cause
      3. Explain the use of execution highlighting for tracing data flow and as a helping tool in conjunction with other debugging tools
      4. Explain and apply breakpoints, execution suspension, and single stepping tools to debug VIs and subVIs
      5. Utilize the Probe tool, indicators, generic probes, and custom probes to display data values
   b. Debugging practices and techniques for different situations
      1. Given an error situation, select the most appropriate method to debug the error
      2. Determine if a given block diagram would result in an error condition

3. **Front panel window and block diagram objects, data types, variables, and software constructs:**
   a-e) Program control, block diagram logic, and data storage for loops, case structures, sequence structures, event structures, and other block diagram structures.
      1. Select and apply the most suitable program control structure
      2. Select and implement a data storage mechanism for a program control structure
      3. Identify and describe the function of looping structure components
      4. Select a While Loop or For Loop as the most appropriate looping structure
      5. Describe auto-indexing and determine the effects of turning indexing on or off with each type of looping structure
      6. Determine the data values in a loop that utilizes auto-indexing, after a set number of iterations occur or upon loop termination
      7. Describe the use and initialization of shift registers as data storage elements
      8. Determine the data values in the shift register(s) after a set number of iterations occur or upon loop termination
      9. Identify the pros and cons and select between a Sequence structure or Case structure
      10. Select the most appropriate data type to wire to the selector terminal of a Case structure
      11. Identify two types of output tunnels in a Case structure and identify the pros and cons of each type
      12. Identify the advantages of Event structures for event-driven programming
      13. Identify the components of an Event structure
      14. Identify the different ways in which an event may be generated
      15. Identify the different events that an Event structure can handle
      16. Recognize the impact of locking the front panel window for user interface events
      17. Identify the difference(s) between local and global variables

4. Programming VIs, functions, and properties

a. General Programming Functions

1. Determine the output or intermediate values of data elements in an application that utilizes VIs and functions from the following list
2. Identify block diagram code that breaks data flow and techniques that enforce data flow
3. Identify block diagram code that can slow down user interface response or update and identify techniques to improve the response
4. Identify race conditions, memory and performance issues associated with the use of local and global variables, Property Nodes, and references.
5. Utilizing VIs and functions from the following list, determine the most appropriate VI(s) or function(s) to complete a specified functionality

List of the VIs and functions that apply this section:

a) Numeric—**Numeric**, **Conversion**, **Data Manipulation**, and **Comparison** palettes
b) Boolean—**Boolean** palette
c) String—**String**, **String / Number Conversion**, and **String /Array /Path Conversion** palettes
d) Path—Path functions on the **File I/O** palette
e) Array functions
f) Cluster—Cluster functions on the **Cluster, Class & Variant** palette
g) Timing—**Timing** palette
h) File I/O—**File I/O** palette
i) Waveform—**Waveform** palette
j) Events—**Events** palette

5. Data Communication and synchronization

a. Data Communication and synchronization

1. Identify the pros and cons of using local or global variables for data communication
2. Select the most appropriate method to communicate data between multiple block diagram sections
3. Determine the output or intermediate values of data elements in an application that utilizes VIs and functions from the following list
4. Utilizing VIs and functions from the following list, determine the most appropriate VI(s) or function(s) that are needed to complete a specified functionality

List of the VIs and functions that apply this section:

a) Notifiers—**Notifier Operations** palette
b) Queues—**Queue Operations** palette
c) Functional Global Variables

6. **Error handling**
   a. Error clusters and wires
      1. Identify the components of error clusters and terminals that accept the error wire.
      2. Identify the differences between errors and warnings
      3. Design VIs that adhere to the *LabVIEW Style Checklist* topic of the *LabVIEW Help*. For example, utilize error checking to control While Loops, handle errors with Case structures, and use appropriate terminals on the connector pane
   b. Error and Dialog VIs
      1. Given a VI or subVI, identify the most appropriate locations to handle errors and notify the user or a calling VI of the error
      2. Utilizing VIs and functions from the **Dialog & User Interface** palette, determine the most appropriate VI(s) or function(s) to complete a specified error handling and reporting functionality
   c. Custom error codes
      1. Identify the numeric range and methods to define custom error codes and use the custom error codes to generate errors from VIs

7. **Design patterns**
   a. Design Patterns:
      1. Identify a design pattern, explain its pros and cons, and compare it with other design patterns
      2. Given an application requirement, select the most appropriate design pattern from the following:
         a) Simple state machine
         b) Event-Based State Machine
         c) Producer/consumer (data)
         d) Producer/consumer (events)
         e) Functional global variable
   b. SubVI Creation
      1. Methods to create subVIs
         a) Identify and explain the methods used to create subVIs and the pros and cons of each method
      2. Connector pane and connection types
         a) Select the most appropriate connector pane and assign terminals according to recommendations in the *LabVIEW Style Checklist* topic of the *LabVIEW Help*
         b) Identify which terminals are Required, Recommended, or Optional
         c) Given a requirement, identify which terminals to set as Required, Recommended, or Optional connections
      3. Polymorphic SubVIs
         a) Evaluate if a polymorphic subVI design is the most appropriate choice

4. Options related to SubVIs
   a) Select and apply the most appropriate execution and window settings for an instance of a SubVI
5. Error handling
   a) Apply error handling to a subVI as recommended in the *LabVIEW Style Checklist* topic of the *LabVIEW Help*

8. VI Server

a. Configuration
   1. Apply appropriate settings for configuring the VI Server
b. References, Property Nodes, Invoke Nodes
   1. Identify uses of reference types, properties, and methods of block diagram objects.
   2. Identify the order of execution of property nodes and methods
   3. Identify the behavior of error handling within and between server nodes.