

# COURSE DESCRIPTION: Ada For Experienced Programmers



Duration: 5-10 days

Format: Lecture/Workshops

Maximum Size: 15

## Overview:

This course is intended to jump-start experienced programmers in the use of the Ada programming language to develop reliable, maintainable software systems. Students will learn how Ada supports software engineering principles, such as abstraction, information hiding, localization, modularity, and re-use, and how to apply these principles in Ada software development. Students will gain experience with the Ada syntax and semantics for data and program structuring, error management, information hiding, object-oriented programming, and generics, tasking, and low-level programming. Finally, we will look at how the new annexes of Ada support domain-specific development: distributed, real-time, and information systems, numeric computation, systems programming, safety, and security.

After this course a student should be able to:

- Understand Ada in the context of an application domain
- Build Ada programs using object-oriented programming
- Write simple concurrent Ada programs
- Use Ada to support software re-use efforts
- Contribute to the design of Ada software applications

In the lectures, extensive examples will be used to illustrate the new features of Ada. In hands-on workshops, the students will practice using the Ada features in typical Ada development environments.

**Target Population:** All software development personnel, including their management and QA engineers, who intend to program in Ada, design for Ada, or review Ada code.

**Prerequisites:** Programming experience, familiarity with a high-level language.

**Materials:** Each student will receive a copy of all lecture materials, lab notes, and reference materials.

## Topics:

- Overview
  - Organization and Rationale
  - Reference Manual and its Annexes
  - Important Differences Between Ada & Other Languages
- Program Organization And Structure
  - Packages
  - Library Units
  - Child Library Units
  - Overview of Object-Oriented Programming in Ada
- Algorithmic Features
  - Control Structures
  - Subprograms
  - Types
    - Strong Typing
    - Scalar Types
    - Constrained Composite Types
    - Unconstrained Arrays
    - Parameterized Types
    - Real Numers
      - Floating Point
      - Fixed Point<sup>1</sup>
      - Decimal<sup>1</sup>
    - Access Types
- Exceptions
- Object-Oriented Programming
  - Program by Extension
  - Polymorphism
  - Dynamic Dispatching
  - Controlled & Limited-Controlled Types
- Ada Predefined Library
- Reuse & Generics
  - Selecting Generic Components to use
  - Instantiating Generic Components
  - Designing Generics Components<sup>1</sup>
- Concurrency and Tasking<sup>1</sup>
  - Basic Concepts
  - Tasks
    - Declaration
    - Implementation
    - Activation
    - Dependency
    - Termination
  - Safe Mutual Exclusion
    - Protected Units & Types
  - Intertask Communication
  - Thinking About Process Timing & Priorities
- Low-Level Features of Ada Language<sup>1</sup>
  - Representation Clauses
  - Interface to Other Languages
  - Unchecked Storage De-allocation
  - Unchecked Type Conversions
- Specialized Annexes<sup>1</sup>
  - Systems Programming
  - Real-Time Systems
  - Distributed Systems
  - Information Systems
  - Numerics
  - Safety & Security
- Conclusions

<sup>1</sup> As appropriate to audience