RUBY PROGRAMMING

DURATION:

4 Days

COURSE DESCRIPTION:

This course covers the fundamental components of the Ruby Programming Language. Emphasis is placed on the object oriented aspects of Ruby. Topics include arrays, hashes, regular expressions, io, exceptions, modules, and applications areas.

WHO SHOULD ATTEND:

This course is intended primarily for those who have programmed in other programming languages such as, but not limited to, C, C++, Java, or Perl.

BENEFITS OF ATTENDANCE:

Upon completion of this course, students will be able to:

- Distinguish and use various Ruby datatypes
- Master the use of arrays and hashes
- Build home grown classes
- Use the extensive pre bundled classes
- Use the I/O facilities of Ruby to read and write binary and text files
- Master the use of Iterators to loop through various data structures
- Use Exceptions in handling various run time errors
- Create Ruby modules
- Use the wide variety of Ruby Modules that come with the Ruby distribution

PREREQUISITES:

Students should have taken the Software Development for Non-Programmers course or have at least six months of programming experience in at least one programming language.

COURSE OUTLINE:

- **CHAPTER 1: AN INTRODUCTION TO RUBY**
    1. What is Ruby?
    2. Installing Ruby
    3. Executing Ruby Code
    4. Getting Help
    5. Dynamic Types
    6. Ruby Reserved Words
    7. Naming Conventions
    8. Comments
- **CHAPTER 2: STANDARD RUBY DATA TYPES**
    1. Numbers
    2. Strings
    3. Simple Input and Output
    4. Converting String Input
    5. Regular Expressions
    6. Time Methods
- **CHAPTER 3: LANGUAGE COMPONENTS**
    1. The `if` Statement
    2. The `case` Construct
    3. Loops
    4. Iterators
    5. Numeric Iterators
    6. String Iterators
    7. Methods
    8. Odds and Ends
- **CHAPTER 4: COLLECTIONS**
    1. Arrays
    2. Array Operator Methods
    3. Array Equality Operator
    4. Arrays as Stacks and Queues
    5. Higher Dimensional Arrays

6. Other Useful Arrays Methods
7. Command Line Arguments
8. Hashes
9. Common Hash Methods
10. Sorting Hashes
11. Iterators with Arrays and Hashes
12. Arrays and Methods
13. Hashes and Methods
14. Named Parameters
15. Symbols
16. Procs
17. Closures

- **CHAPTER 5: CLASSES**

1. Objects
2. Brief History of OOP
3. OOP Vocabulary
4. Creating a New Class
5. Using Objects
6. Defining Operator Methods
7. Inheritance
8. Ancestors
9. `self`
10. Access Levels - `public`
11. Access Levels – `private`
12. Access Levels - `protected`
13. Access Levels - Specification
14. Class Data and Class Methods
15. Adding Methods to Classes and Objects
16. Special Global Variables
17. Scope of Variables
18. Built-in Classes
19. The `Math` Class
20. The `NilClass` Class
21. `TrueClass` and `FalseClass`
22. Built-in Class Hierarchy

- **CHAPTER 6: INPUT AND OUTPUT**

1. Introduction
2. Reading from the Standard Input
3. Reading a Character at a Time
4. Writing to the Standard Output
5. Reading and Writing Disk Files
6. Reading Files Using Iterators
7. I/O With Command Line Commands
8. Seeking About Files
9. `tell`
10. Capturing Data About Files
11. Processing Directories

- **Chapter 7: Exceptions**
  1. Introduction
  2. Exception Hierarchy
  3. Handling Exceptions
  4. Multiple Rescue Clauses
  5. Exceptions are Classes
  6. `ensure`
  7. `retry`
  8. `raise`
  9. Creating Your Own Exceptions
  10. `catch` and `throw`
- **Chapter 8: Modules**
  1. Introduction
  2. Using Core Ruby Classes
  3. Ruby Standard Library
  4. `require`
  5. Search Path
  6. File Organization
  7. `load`
  8. Modules
  9. `include`
  10. Mixins
  11. Using the `Comparable` Module
  12. Collection Classes
  13. `yield`

14. Using the Enumerable Module

- **Chapter 9: Odds and Ends**

1. Ruby Conventions
2. Bit Manipulation
3. Substituting
4. Marshalling
5. Reflection
6. `grep`
7. Classes are Objects
8. Aliasing
9. Testing
10. `Test::Unit::TestCase`
11. Testing Your Own Classes
12. Freezing Objects

sbmadake@gmail.com +919922292223