College of San Mateo
Official Course Outline

1. **COURSE ID:** CIS 113 **TITLE:** Ruby Programming
   Units: 4.0 units **Hours/Semester:** 48.0-54.0 Lecture hours; 48.0-54.0 Lab hours; and 96.0-108.0 Homework hours
   Method of Grading: Grade Option (Letter Grade or Pass/No Pass)
   Recommended Preparation:
   Completion of CIS 111 or CIS 254.

2. **COURSE DESIGNATION:**
   Degree Credit
   Transfer credit: CSU; UC

3. **COURSE DESCRIPTIONS:**
   Catalog Description:
   Comprehensive course in Ruby, an open-source dynamic object-oriented scripting language. Covers variables, arrays and hashes, methods and procs, classes, objects, and writing server-side Ruby scripts for the Web. Also covered are exception handling, regular expressions, I/O objects, and modules. An introduction to SQL and the MySQL database, and advanced topics such as Model-View-Controller architecture and agile Web application development with the Ruby on Rails framework. Intended for students with previous programming experience.

4. **STUDENT LEARNING OUTCOME(S) (SLO'S):**
   Upon successful completion of this course, a student will meet the following outcomes:
   1. Develop server-side Ruby scripts for publishing on the Web.
   2. Employ control structures, methods, procs, arrays and hashes to create Ruby programs.
   3. Explain object-oriented programming and input/output processing and apply these concepts to develop dynamic interactive Ruby applications.
   4. Discuss Model-View-Controller architecture and its relationship to Ruby on Rails applications.
   5. Use SQL commands and the MySQL database together with Ruby.
   6. Create an advanced project using MySQL, Ruby and the Ruby on Rails framework.

5. **SPECIFIC INSTRUCTIONAL OBJECTIVES:**
   Upon successful completion of this course, a student will be able to:
   1. Develop server-side Ruby scripts for publishing on the Web.
   2. Employ control structures, methods, procs, arrays and hashes to create Ruby programs.
   3. Explain object-oriented programming and input/output processing and apply these concepts to develop dynamic interactive Ruby applications.
   4. Discuss Model-View-Controller architecture and its relationship to Ruby on Rails applications.
   5. Use SQL commands and the MySQL database together with Ruby.
   6. Create an advanced project using MySQL, Ruby and the Ruby on Rails framework.

6. **COURSE CONTENT:**
   Lecture Content:
   1. Introduction
      A. History of Ruby
      B. Dynamic Web Applications
      C. Ruby Installation
      D. Web Servers and Server-Side Programming
      E. Ruby and Databases
      F. UNIX/Linux Environment
   2. Data and Operations
      A. Data and Types
      B. Variables
      C. Constants
      D. Operators
      E. Expressions
      F. Operator Precedence

G. Scope
3. Conditional Statements and Control Structures
   A. If Else
   B. Case
   C. While
   D. Do
   E. Foreach
   F. For In
   G. Redo
4. Methods
   A. Methods Definition
   B. Methods and Blocks
   C. Calling a Method
   D. Method Parameters
   E. Method Return Values
   F. Exception Handling
5. Arrays and Hashes
   A. Indexing Arrays
   B. Initializing Arrays
   C. Array Class
   D. Hashes
   E. Blocks and Iterators
6. Classes and Objects
   A. Defining a Class
   B. Objects and Attributes
   C. Access Control
   D. Inheritance and Messages
   E. Class Variables and Class Methods
   F. Modules and Namespaces
   G. Mixins
7. Web Interaction and File Access
   A. Ruby HTML Code Generation
   B. CGI Code Generation
   C. Templating Systems
   D. Form Input
   E. Validation and Regular Expressions
   F. Cookies
   G. Sessions
   H. Reading/Writing Files
8. SQL and the MySQL Database
   A. Introduction to SQL Syntax
   B. Designing and Creating a Table in MySQL
   C. MySQL Data Types
9. Dynamic Applications with Ruby on Rails and MySQL
   A. Model-View-Controller Architecture
   B. Connecting to MySQL
   C. Active RecordAction Controller
      a. Object-Relational Mapping
   D. Action View
   E. Authentication
   F. Testing and Debugging

Lab Content:

Write Ruby applications using variables, data types, strings and methods.
Write Ruby applications using loops, arrays. hashes, blocks and sorting.
Build Rails applications using Rails best practices.
Manage a database with migrations.
Build complex models using ActiveRecord, including associations, validations and callbacks.
Programming projects to upgrade the UI using Rails' built-in support for JavaScript and Ajax.

Programming project to add functionality and extend a Rails application using third-party plugins and gem libraries.
Employ TDD (Test Driven Development) and use Rails to write tests to validate an application's behavior.

7. **REPRESENTATIVE METHODS OF INSTRUCTION:**
Typical methods of instruction may include:
   A. Lecture
   B. Lab
   C. Directed Study
   D. Discussion
   E. Other (Specify): The course will include the following instructional methods as determined appropriate by the instructor: • Lecture will be used to introduce new topics; • Teacher will model problem-solving techniques; • Class will solve a problem together, each person contributing a potential "next step"; • Students will participate in short in-class projects (in teacher-organized small groups) to ensure that students experiment with the new topics in realistic problem settings; • Teacher will invite questions AND ANSWERS from students, generating discussion about areas of misunderstanding; • Teacher will create and manage an Internet conference for discussion of course topics; and • Students will work in small groups to solve programming assignments.

8. **REPRESENTATIVE ASSIGNMENTS**
Representative assignments in this course may include, but are not limited to the following:
Writing Assignments:
Weekly programming assignments. Assignments are related to course content and cover object-oriented programming, server-side scripting, control structures, methods, procs, arrays, hashes, SQL commands and databases, and Model-View-Controller architecture using Rails.
Reading Assignments:
Reading assignments accompanied by self-test questions and running coding samples.

9. **REPRESENTATIVE METHODS OF EVALUATION**
Representative methods of evaluation may include:
   A. Class Participation
   B. Class Work
   C. Exams/Tests
   D. Group Projects
   E. Homework
   F. Lab Activities
   G. Projects
   H. Quizzes
   I. Written examination

10. **REPRESENTATIVE TEXT(S):**
Possible textbooks include:
   A. Black, D., Leo, J. *The Well Grounded Rubyist*, 3rd ed. Manning Publications, 2019
   B. Hartl, M. *Ruby on Rails Tutorial*, 6th ed. Addison-Wesley Professional, 2020
   C. DiLeo, C. & Cooper, P. *Beginning Ruby 3: From Beginner to Pro*, 4th ed. Apress, 2021

Origination Date: October 2020
Curriculum Committee Approval Date: November 2020
Effective Term: Fall 2021
Course Originator: Melissa Green