**CMSC330 Ruby Basics (Days 1 and 2)**

**Fall 2019**

After this unit you should be able to:

I. Give a personal explanation of how to learn a programming language

A. Find out who, what, why and when- it helps to know goals and attributes

B. Open up reference documentation so you have that at hand

C. Don't just follow tutorials – explore on your own to

D. Do small examples to understand how statements and features work

E. Do larger examples to exercise your understanding

II. Find, install and run Ruby programs

A. Use quick, everywhere online IDEs

<https://ruby.github.io/TryRuby/>

<https://repl.it/languages/ruby>

B. Install the official distribution of Ruby, and the current version (use version from TAs)

<https://www.ruby-lang.org/en/downloads/>

C. Run fragments of Ruby on the command line in the Interactive Ruby environment

$ irb

D. Run single and multi-file Ruby programs using the standard interpreter

$ ruby -w program.rb

E. Run Ruby programs as scripts

Add #!/usr/bin/env ruby as first line; make executable; run from cmd line

II. Find and use official and other Ruby documentation tutorial

A. Find general Ruby resources (see readings at the end for more)

B. Find, navigate and read the official reference documentation

<https://docs.ruby-lang.org/en/2.6.0/>

<https://ruby-doc.org/core-2.6.3/>

III. List and explain basic attributes of Ruby

A. Imperative, object oriented

B. Scripting: quick, short programs integrated with system calls, often text oriented

C. Dynamic: variable typing and other program elements determined/extended at run time

D. Implicit declarations: variables get types implicitly from values, not from type names

E. Dynamic type checking: types are set during run time and can change

F. Pure OO: everything is an object

G. Lightweight syntax: syntax elements are optional (eg, semicolons, parens)

\*\*You don’t need to memorize the origins of Ruby, but knowing the who, what and why of a

programming language helps understand its goals and characteristics.

Designed by Yukihiro Matasumoto in 1993-1995 because he wanted a pure 00, easy,

scripting language with a lightweight syntax.

ISO standard created in 2010-2011- maintained by standards committee

IV. Read and write basic, short Ruby programs and fragments with numbers, strings and output

(Objective: start Ruby by comparing to Java)

A. Create and assign values to variables with implicit declarations and dynamic typing

B. Use puts and print to output numeric and string values

C. Read and write basic Ruby imperative control structures, with correct syntax/semantics

If then, if else, if elsif, plus if as modifier after statement

unless then, plus unless as modifier after statement

while loop, plus until loop

case-when statement

D. Note differences in syntax and semantics from Java on these language constructs

Dynamic vs. static typing, implicit vs. explicit declaration

Required use of end to terminate control structure

Extra control structures like if-modifier, unless

Optional use of parens on method calls, semicolons ; on stmts, then in control

Difference from C/Java that condition is satisfied unless value is nil or false (0 not false)

E. Compare and contrast Ruby, Java and C, and give the pros and cons for these constructs

V. Use the basics of Ruby objects, both syntax and semantics, to read and write short programs

A. Explain what it means that Ruby is “pure” OOP – everything is an object, including classes

B. Identify class names as beginning with capital letter (so are constant identifiers)

C. Use the following classes: Integer, Float, String, Symbol, NilClass, Class, Object

D. Use the following methods available on all objects;

obj.class (get class), obj.methods (get available methods), obj.to\_s, obj.inspect

E. Instantiate, concatenate, compare and otherwise operate on Strings

String.new, s.length, s1+s2, s1 == s2, s.chomp, s.chomp!, interpolation using #{exp}

Here-doc for strings on multiple lines

F. Know the method convention of *name?* for Boolean inquiry, *name!* for modifying object G. Identify symbols as :identifier or :”string”; know their basic behavior (unique objects)

VI. Use Ruby arrays

A. Use basic features: heterogeneous, recursive, indexed from zero, grow and shrink

B. Write and read Array literals, including empty array []; create with Array.new

C. Use the following Array methods:

:length, :member?, :delete\_at, :delete, :sort, :index, :sort, :uniq

D. Use Arrays as stacks and queues using :push, :pop, :shift, :unshift (and :insert)

(Using with code blocks and enumerators later)

VII. Use Ruby Hash tables

A. Create Hash tables with Hash literals, with Hash.new, and with Hash.new(default)

B. Retrieve value from table with h[key]; retrieve key with h.key(value); insert/update value

C. Test for key with h.key?, for value with h.value? (h.has\_key? is deprecated)

D. Delete key/value pairs with delete\_(key)

E. Get Arrays of keys or values with :values and :keys

(Ditto on using code blocks and enumerators)

**Suggested readings:**

There’s a number of introductory tutorials on Ruby that you might use. Some do more than we need; some do less. It’s up to you to pick one or more, work through, and focus on what we ask you to know.

The Ruby about page gives an overview of the nature and history of Ruby. Good quick read.

<https://www.ruby-lang.org/en/about/>

The official Ruby web site has a page of documentation options.

<https://www.ruby-lang.org/en/documentation/>

A safe good start would be the Pickaxe Book by Thomas and Hunt.

<https://ruby-doc.com/docs/ProgrammingRuby/>

If you need amusement and have some time, see <http://poignant.guide/book/chapter-1.html>

For a basic but slow start, see: <https://pine.fm/LearnToProgram/>

The Wikibooks tutorial at

<https://en.wikibooks.org/wiki/Ruby_Programming>

goes into more than we need – more details and options for strings, numbers, and much more. However, it is terse and to the point, and if you need the details, good.

The RubyLearning guide at is similar.

<http://rubylearning.com/satishtalim/tutorial.html>

The official documentation at

<https://ruby-doc.org/core-2.6.3/index.html>

is not the easiest to navigate, but if you find useful pages, it’s good.