CompuScholar, Inc.

Alignment to
"*ICT – Programming & Logic Essentials*"
Certification Exam Requirements

ICT Course Details:

Course Name:	ICT – Programming & Logic Essentials
Course Code(s):	ICT – Programming & Logic Essentials
Standards Link:	http://www.ictcertified.com/ict-essentials/programming-and-logic.php

CompuScholar Course Details:

Course Title:	Java Programming (Abridged)
Course ISBN:	978-0-9887070-4-7
Course Year:	2015

OR

Course Title:	Java Programming (AP)
Course ISBN:	978-0-9887070-2-3
Course Year:	2015

Note 1: Citation(s) listed may represent a subset of the instances where objectives are met throughout the

Introduction

The ICT – Programming & Logic Essentials exam is designed to test basic knowledge structured programming in any high-level language. Preparation for this exam can generally be accomplished within a single school year.

EITHER the CompuScholar "TeenCoder: Java Programming (Abridged)" OR the "TeenCoder: Java Programming" courses can be used to prepare for this exam. All exam topics are covered by both courses.

Exam Requirements

Sub-Domain 4.1 - Demonstrate fundamental knowledge of programming languages and how they are used to communicate with computers.	CITATION(S)
4.1.1 Define “programming,” and discuss its role in computing.	Chapter 1, Lesson 3
4.1.2 Explain the binary representation of data and programs in computers.	Chapter 4, Lesson 1 Chapter 17, Lesson 2
4.1.3 Distinguish among the three types of programming languages (machine, assembly, high-level), and give examples.	Chapter 1, Lesson 3

4.1.4 Compare and contrast languages that are usually compiled (e.g., C++, Java) and interpreted (e.g., JavaScript, Python).	Chapter 1, Lesson 3
4.1.5 Describe the structure of a simple program, and explain why sequencing is important.	Chapter 2, Lesson 2 Chapter 7, Lesson 2
4.1.6 Write a program design document using pseudo-code that shows program flow.	Chapter 17, Lesson 4 Team Project, Lesson 2

Sub-Domain 4.2 - Demonstrate the use of logic and problem-solving, and relate these concepts to computer programming.	**CITATION(S)**
4.2.1 Explain strategies used in problem-solving, and relate them to computer programming.	Chapter 17, Lesson 4
4.2.2 Define the term “algorithm,” and explain how it relates to problem-solving.	Chapter 17, Lesson 4
4.2.3 Explain the three types of programming errors (i.e., logic, syntax, runtime), and describe the forms of testing that can be used to locate and debug errors.	Chapter 9, Lesson 1 Chapter 9, Lesson 3
4.2.4 Solve a problem using logic by planning a strategy, designing and testing a hypothesis, and/or creating a set of step-by-step instructions to perform a task.	Team Project, Lessons 1 & 2 Supplemental Lesson 4

Subdomain 4.3 - Demonstrate knowledge of fundamental structured programming concepts.	**CITATION(S)**
4.3.1 Define “structured programming,” and discuss the advantages of this approach.	Chapter 8, Lesson 1
4.3.2 Define the three main programming control structures used in structured programming: sequential, selection (decision), and iteration (loops).	Chapter 7, Lesson 2 Chapter 7, Lesson 3 Chapter 7, Lesson 4
4.3.3 Describe iterative programming structures (e.g., while, do/while, etc.) and how they are used in programming.	Chapter 7, Lesson 4 Chapter 7, Lesson 5
4.3.4 Describe selection programming structures (e.g., if/then, else) and explain the logic used for if statements.	Chapter 7, Lesson 2 Chapter 7, Lesson 3
4.3.5 Write a simple program in pseudo-code that uses structured programming to solve a problem.	Chapter 17, Lesson 4

Subdomain 4.4 - Demonstrate proficiency in basic programming and working with data.	**CITATION(S)** **NOTE: This course uses the Eclipse IDE as a high-level programming environment.**
4.4.1 Explain the types and uses of variables in programming.	Chapter 4, Lesson 1 Chapter 4, Lesson 2
4.4.2 Explain basic object-oriented concepts.	Chapter 10 (all Lessons) Chapter 11 (all Lessons) Chapter 15 (all Lessons)

4.4.3 Describe fundamental Boolean concepts, including Boolean algebra, operators, logic.	Chapter 7, Lesson 1
4.4.4 Create animated objects using a high-level programming environment (e.g., Alice, Greenfoot) to control their behavior.	*Chapter 20 (Abridged) OR *Chapter 21 (full) (vector graphics)
4.4.5 Create a simple program that uses animated objects.	*Chapter 20 (Abridged) OR *Chapter 21 (full) (vector graphics)
4.4.6 Convert a simple program from pseudo-code into a common high-level programming environment (e.g., Alice, Greenfoot).	* Chapter 17, Lesson 4 and all programming activities
4.4.7 Create a simple program using a high-level programming environment (e.g., Alice, Greenfoot).	* Simple to complex programs are created in every chapter.
4.4.8 Troubleshoot and debug errors in code	Chapter 9 (all Lessons)