CompuScholar, Inc.

Alignment to the College Board AP **Computer Science A** Standards

9th - 12th grades

AP Course Details:

Course Title:	AP Computer Science A
Grade Level:	9th - 12th grades
Standards Link:	ap-computer-science-a-course-description.pdf

CompuScholar Course Details:

Course Title:	Java Programming (AP)
Course ISBN:	978-0-9887070-2-3
Course Year:	2017

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

Note 2: Citation(s) for a "Lesson" refer to the "Lesson Text" elements and associated "Activities" within the course, unless otherwise noted. The "Instructional Video" components are supplements designed to introduce or re-enforce the main lesson concepts, and the Lesson Text contains full details.

AP Course Description

This course teaches students the fundamentals of the Java programming language and covers all required topics defined by the College Board's AP Computer Science A course description.

AP Lab Requirements

The AP Computer Science A course must include a minimum of 20 hours of hands-on structured-lab experiences to engage students in individual or group problem solving.	**CITATION(S)**
This course easily meets and exceeds the 20-hour minimum lab requirement with hands-on lesson exercises and chapter activities. In addition, coverage of both new example labs and the old GridWorld lab is provided for teachers to use as needed.	See "Work with Me" sections within lessons and "Chapter Activities" in each chapter.
Magpie Lab (recommended starting in 2014-2015)	Chapter 26, Lesson 1
Picture Lab (recommended starting in 2014-2015)	Chapter 26, Lesson 2
Elevens Lab (recommended starting in 2014-2015)	Chapter 26, Lesson 3
GridWorld Case Study (no longer required, but available for use if desired)	Chapter 27

AP Topic Outline

I. Object-Oriented Program Design	CITATION(S)
A. Program and Class Design	
1. Problem analysis	Covered in all Chapter Activities
2. Data abstraction and encapsulation	Chapter 10, Lesson 1 Chapter 10, Lesson 3
3. Class specifications	Chapter 10, Lesson 2
3. Class interface specifications	Chapter 11, Lesson 2
3. Class relationships	Chapter 10, Lesson 2
3. Class extension using inheritance.	Chapter 15, Lesson 2
4. Code reuse	Chapter 10, Lesson 1
5. Data representation and algorithms	Chapter 4, Lesson 2 Chapter 20, Lesson 1
6. Functional decomposition	Chapter 8, Lesson 1 Chapter 22, Lesson 1

II. Program Implementation	CITATION(S)
A. Implementation Techniques	
1. Top-Down development	Chapter 24, Lesson 1
2. Bottom-up development	Chapter 24, Lesson 1
3. Object-oriented	Chapters 10, 11, 15, 16
4. Encapsulation and information hiding	Chapter 10, Lesson 1 Chapter 10, Lesson 3
5. Procedural abstraction	Chapter 8 Chapter 10, Lessons 1 - 2
B. Programming Constructs	
1. Primitive data types vs. reference types	Chapter 4, Lesson 1 Chapter 5, Lesson 1
2.a. Constant declarations	Chapter 4, Lesson 2

2.b. Variable declarations	Chapter 4, Lesson 2
2.c. Methods and parameters	Chapter 8, Lesson 1 Chapter 8, Lesson 2
2.d. Class declarations	Chapter 10, Lesson 2
2.e. Interface declarations	Chapter 11, Lesson 2
3. Text output using System.out.print and System.out.println	Chapter 4, Lesson 3
4.a. Method call control	Chapter 8, Lesson 3
4.b. Sequential execution control	Chapter 2, Lesson 2 Chapter 7, Lesson 2
4.c. Conditional execution control	Chapter 7, Lesson 2
4.d. Iteration control	Chapter 14, Lesson 3
4.e. Recursion control	Chapter 19, Lesson 1
5.a. Numeric expression evaluation	Chapter 4, Lesson 2
5.b. String expression evaluation	Chapter 5, Lesson 2 Chapter 5, Lesson 3
5.c. Boolean expressions, short-circuit evaluation, De Morgan's law	Chapter 7, Lesson 1
C. Java Library Classes and Interfaces in the AP Java Subset	See cross-reference tables below

III. Program Analysis	**CITATION(S)**
A. Testing	
1. Development of appropriate test cases, including boundary cases	Chapter 9, Lesson 3 Chapter 24, Lesson 3
2. Unit testing	Chapter 9, Lesson 3 Chapter 24, Lesson 3
3. Integration testing	Chapter 9, Lesson 3 Chapter 24, Lesson 3
B. Debugging	
1. Error categories: compile-time, run-time, logic	Chapter 9, Lesson 1
2. Error identification and correction	Chapter 9, Lesson 3
3. Techniques such as using a debugger, adding extra output statements, or hand-tracing code	Chapter 9, Lesson 3 Chapter 9, Lesson 4

C. Runtime Exceptions	Chapter 9, Lesson 1
D. Program Correctness	
1. Pre- and post- conditions	Chapter 24, Lesson 3
2. Assertions	Chapter 24, Lesson 3
E. Algorithm Analysis	
1. Statement execution counts	Chapter 20, Lesson 3
2. Informal running time comparison	Chapter 20, Lesson 2
F. Numerical Representations of Integers	
1. Representations of non-negative integers in different bases	Chapter 17, Lesson 2
2. Implications of finite integer bounds	Chapter 17, Lesson 2

IV. Standard Data Structures	**CITATION(S)**
A. Primitive data types (int, boolean, double)	Chapter 4, Lesson 1
B. Strings	Chapter 5 (all lessons)
C. Classes	Chapters 10 and 11 (all lessons)
D. Lists	Chapter 14, Lesson 2
E. Arrays (1-dimensional and 2-dimensional)	Chapter 14, Lesson 1

V. Standard Operations and Algorithms	**CITATION(S)**
A. Operations on Data Structures	
1. Traversals	Chapter 14, Lesson 3
2. Insertions	Chapter 14, Lesson 2
3. Deletions	Chapter 14, Lesson 2
B. Searching	
1. Sequential	Chapter 19, Lesson 3

2. Binary	Chapter 19, Lesson 3
C. Sorting	
1. Selection	Chapter 19, Lesson 2
2. Insertion	Chapter 19, Lesson 2
3. Mergesort	Chapter 19, Lesson 2

VI. Computing in Context	**CITATION(S)**
A. System Reliability	Chapter 1, Lesson 4
B. Privacy	Chapter 1, Lesson 4
C. Legal issues and intellectual property	Chapter 1, Lesson 4 Chapter 1, Lesson 5
D. Social and ethical ramifications of computer use	Chapter 1, Lesson 4

Cross-Reference Tables

Java AP Subset - Langauge Features and Other Testable Topics	**CITATION(S)**
Comments /* */, //, and /** */, Javadoc @param and @return comment tags	Chapter 2, Lesson 2 Chapter 24, Lesson 2
Primitive Types int, double, boolean	Chapter 4, Lesson 1
Operators: Arithmetic: +, -, *, / and %	Chapter 4, Lesson 2
Operators: Arithmetic: Increment/Decrement: (++), (--)	Chapter 4, Lesson 2
Operators: Assignment: =, +=, -=, *=, /=, and %=	Chapter 4, Lesson 2
Operators: Relational: ==, !=, <, >, <=, >=	Chapter 7, Lesson 1
Operators: Logical: !, &&, ||	Chapter 7, Lesson 1
Operators: Numeric casts: (int), (double)	Chapter 4, Lesson 2
Operators: String concatenation	Chapter 5, Lesson 4

Object Comparison: object identity (==, !=) vs. object equality (equals)	Chapter 5, Lesson 2 Chapter 7, Lesson 1 Chapter 15, Lesson 5
Object Comparison: String compareTo	Chapter 5, Lesson 3
Escape Sequences \", \\, \n inside strings	Chapter 4, Lesson 3
Input / Output System.out.print, System.out.println	Chapter 4, Lesson 3
Exceptions: ArithmeticException, NullPointerException, IndexOutOfBoundsException, ArrayIndexOutOfBoundsException, IllegalArgumentException.	Chapter 9, Lesson 1 Chapter 9, Lesson 2
Arrays: 1-dimensional and 2-dimensional rectangular arrays	Chapter 14, Lesson 1
Arrays: initializer list: { ... }	Chapter 14, Lesson 1
Arrays: row-major order of 2-dimensional array elements	Chapter 14, Lesson 1
Control Statements: if, if/else	Chapter 7, Lesson 2
Control Statements: while loops	Chapter 7, Lesson 5
Control Statements: for loops	Chapter 7, Lesson 4
Control Statements: enhanced for (for-each)	Chapter 14, Lesson 3
Control Statements: return	Chapter 7, Lesson 2 Chapter 8, Lesson 2
Variables: parameter variables	Chapter 8, Lesson 2
Variables: local variables	Chapter 10, Lesson 2
Variables: private instance variables	Chapter 10, Lesson 3
Variables: visibility (private)	Chapter 10, Lesson 3
Variables: static (class) variables	Chapter 11, Lesson 3
Variables: visibility (public, private)	Chapter 10, Lesson 3
Variables: final	Chapter 4, Lesson 2
Methods: visibility (public, private)	Chapter 10, Lesson 3

Methods: static, non-static	Chapter 10, Lesson 2 Chapter 11, Lesson 3
Methods: signatures	Chapter 8, Lesson 2
Methods: overloading	Chapter 8, Lesson 2
Methods: overriding	Chapter 15, Lesson 4
Methods: parameter passing	Chapter 8, Lesson 2
Constructors super(), super(args)	Chapter 15, Lesson 6
Classes: new	Chapter 10, Lesson 2
Classes: visibility (public)	Chapter 10, Lesson 3
Classes: accessor methods	Chapter 10, Lesson 3
Classes: modifier (mutator) methods	Chapter 10, Lesson 3
Classes: Design/create/modify classes	Chapters 10, 11, 15, 16, 25, 26 (All lessons and activities)
Classes: Create subclass of a superclass (abstract, non-abstract)	Chapters 15 Lesson 2
Classes: Create class that implements interface	Chapter 11, Lesson 2
Interfaces: Design/create/modify an interface	Chapter 11, Lesson 2 Chapter 11 Activity Chapter 26, Activity 2 and 3
Inheritance: Understand inheritance hierarchies	Chapter 15, Lesson 2 Chapter 15, Lesson 3
Inheritance: Design/create/modify subclasses	Chapter 15, Lesson 2
Inheritance: Design/create/modify classes that implement interfaces	Chapter 11, Lesson 2 Chapter 15, Lesson 2
Packages: import packageName.className	Chapter 2, Lesson 4
Miscellaneous OOP: "is-a" and "has-a" relationships	Chapter 10, Lesson 2
Miscellaneous OOP: null	Chapter 5, Lesson 1
Miscellaneous OOP: this	Chapter 10, Lesson 2
Miscellaneous OOP: super.method(args)	Chapter 15, Lesson 6

Standard Java Library	See Quick-Reference Below
Notes	
1. Students are expected to understand the operator precedence rules of the listed operators.	Chapter 7, Lesson 1
2. The increment/decrement operators ++ and -- are part of the AP Java subset.	Chapter 4, Lesson 2
3. Students need to understand the “short circuit” evaluation of the && and || operators.	Chapter 7, Lesson 1
4. Students are expected to understand “truncation towards 0” behavior as well as the fact that positive floating-point numbers can be rounded to the nearest integer as (int)(x + 0.5), negative numbers as (int)(x - 0.5).	Chapter 4, Lesson 2
5. String concatenation + is part of the AP Java subset. Students are expected to know that concatenation converts numbers to strings and invokes toString on objects.	Chapter 5, Lesson 4
6. User input is not included in the AP Java subset.	Though not required, the course covers both console input (Chapter 6) and GUI/Swing (Chapters 12, 13)
7. Both arrays of primitive types (e.g., int[], int[][]) and arrays of objects (e.g., Student[], Student[][]) are in the subset.	Chapter 14, Lesson 1
8. Students need to understand that 2-dimensional arrays are stored as arrays of arrays. For the purposes of the AP CS A Exam, students should assume that 2-dimensional arrays are rectangular (not ragged) and the elements are indexed in row-major order. Students are expected to be able to access a row of a 2-dimensional array, assign it to a 1-dimensional array reference, pass it as a parameter, and use loops (including for-each) to traverse the rows.	Chapter 14, Lesson 1
9. The main method and command-line arguments are not included in the subset. In free-response questions, students are not expected to invoke programs.	main() is covered in Chapter 2, Lesson 2
10. Students are required to understand when the use of static methods is appropriate.	Chapter 11, Lesson 3
11. If a subclass constructor does not explicitly invoke a superclass constructor, the Java compiler automatically inserts a call to the no-argument constructor of the superclass.	Chapter 15, Lesson 6
12. Students are expected to implement constructors that initialize all instance variables.	Chapter 11, Lesson 1
13. Students are expected to write interfaces or class declarations when given a general description of the interface or class.	Chapters 10, 11, 15, 16, 26

14. Students are expected to extend classes and implement interfaces. Students are also expected to have knowledge of inheritance that includes understanding the concepts of method overriding and polymorphism. Students are expected to implement their own subclasses. Students are expected to read the definition of an abstract class and understand that the abstract methods need to be implemented in a subclass. Students are similarly expected to read the definition of an interface and understand that the abstract methods need to be implemented in an implementing class.	Chapters 11, 15, 16, 26
15. Students are expected to understand that conversion from a subclass reference to a superclass reference is legal and does not require a cast.	Chapter 15, Lesson 3
16. The use of this is restricted to passing the implicit parameter in its entirety to another method (e.g., obj.method (this)) and to descriptions such as “the implicit parameter this".	Chapter 10, Lesson 2
17. The use of generic collection classes and interfaces is in the AP Java subset, but students need not implement generic classes or methods.	Chapter 14, Lesson 2
18. Students are expected to know a subset of the constants and methods of the listed Standard Java Library classes and interface	See Java Quick Reference table below

Java Quick Reference	**CITATION(S)**
class java.lang.Object	Chapter 15, Lesson 5
.....boolean equals(Object other)	Chapter 15, Lesson 5
.....String toString()	Chapter 15, Lesson 5
class java.lang.Integer	Chapter 4, Lesson 2
.....Integer(int value)	Chapter 4, Lesson 2
.....int intValue()	Chapter 4, Lesson 2
.....Integer.MIN_VALUE	Chapter 4, Lesson 2
.....Integer.MAX_VALUE	Chapter 4, Lesson 2
class java.lang.Double	Chapter 4, Lesson 2
.....Double(double value)	Chapter 4, Lesson 2

.....double doubleValue()	Chapter 4, Lesson 2
class java.lang.String	Chapter 5, Lesson 1
.....int length()	Chapter 5, Lesson 3
.....String substring(int from, int to)	Chapter 5, Lesson 3
.....String substring(int from)	Chapter 5, Lesson 3
.....int indexOf(String str)	Chapter 5, Lesson 3
.....int compareTo(String other)	Chapter 5, Lesson 3
class java.lang.Math	Chapter 17, Lesson 1
.....static int abs(int x)	Chapter 17, Lesson 1
.....static double abs (double x)	Chapter 17, Lesson 1
.....static double pow(double base, double exponent)	Chapter 17, Lesson 1
.....static double sqrt(double x)	Chapter 17, Lesson 1
.....static double random()	Chapter 17, Lesson 1
interface java.util.List<E>	Chapter 14, Lesson 2
.....int size()	Chapter 14, Lesson 2
.....boolean add(E obj)	Chapter 14, Lesson 2
.....void add(int index, E obj)	Chapter 14, Lesson 2
.....E get(int index), set(int index, E obj), remove(int index)	Chapter 14, Lesson 2
class java.util.ArrayList<E> implements java.util.List<E>	Chapter 14, Lesson 2