AP COMPUTER SCIENCE A

Credits: 0.5

COURSE OVERVIEW AND GOALS

This one-semester course is intended to introduce you to the concepts of computer programming. This course has 20 lessons organized into four units, plus four Unit Activities. Each lesson contains one or more Lesson Activities.

In AP Computer Science A, you will describe the basic concepts of computer programming. You will compile and run a simple Java program. You will use arithmetic, relational, and logical operators. You will implement algorithms, and use different types of loop and decision- making statements. You will create and use classes. You will create and manipulate one- dimensional and two-dimensional arrays. You will perform sequential search, binary search, selection sort, and insertion sort on an array. You will explain and implement object-oriented programming design. You will implement inheritance, polymorphism, and abstraction. Further, you will describe privacy and legality in the context of computing.

Your teacher will grade your work on the Unit Activities, and you will grade your work on the Lesson Activities by comparing them with the given sample responses. The Unit Activities (submitted to the teacher) and the Lesson Activities (self-checked) are major components of this course. There are other assessment components, namely the mastery test questions that feature along with the lesson; the pre- and post-test questions that come at the beginning and end of the unit, respectively; and an end-of-semester test. All of these tests are a combination of simple multiple-choice questions and technology-enhanced (TE) questions.

By the end of this course, you will be able to do the following:

- Describe the basic concepts of computer programming and compile a simple Java program.
- Identify, describe, and employ Java variables and data types.
- Describe and use arithmetic, relational, and logical operators.
- Describe and use different types of loop and decision-making statements.
- Describe, create, and use classes and methods.
- Create String objects, ArrayList objects, and one-dimensional and two-dimensional

arrays.

- Perform selection and insertion sort on an array and compare their performance.
- Perform sequential and binary searches on an array and compare their performance.
- Explain and implement object-oriented programming design.
- Explain and implement inheritance, polymorphism, and abstraction.
- Create an interface.
- Describe privacy and legality in the context of computing.

Course Components and Grading Rubric

The table gives a breakdown of the weight for each component in the course. Weight represents the percentage of the total score coming from each activity.

Course Components	Count	Weight
Pretests. *Pretests are optional assessments, typically designed for credit recovery use. If a student shows mastery of a lesson's objective, the student may be automatically exempted from that lesson in the upcoming unit. Typically, teachers do not choose to employ exemptive pretests for first-time credit courses. Pretests are not included as a component of the student's final grade.*	4	0%
Module. *Each module in this course contains an interactive tutorial and an associated mastery test. Tutorials may include one or more Lesson Activities that constitute tasks associated with the tutorial. This course also includes practice questions to prepare students for the mastery test. The module score comes from a student's score on the mastery test.*	20	20%
Discussion. *Online discussions allow for higher-order thinking about terminal objectives. An online threaded discussion mirrors the educational experience of a classroom discussion. Teachers can initiate a discussion by asking a complex, open-ended question. Students can engage in the discussion by responding both to the question and to the thoughts of others. Each unit in a course has one predefined discussion topic; teachers may add more discussion topics.*	4	20%
Unit Activity. *Unit Activities are at the end a unit and constitute one or more small tasks. Their purpose is to deepen understanding of key unit concepts and tie them together. Each Unit Activity includes a simple rubric. The teacher versions include both a rubric and modeled sample answers. Unit Activities are teacher graded.*	4	20%
Posttest. *The posttest appears at the end of the unit and mirrors the pretest in structure, content, and complexity.*	4	20%

Course Components	Count	Weight
End of Semester Test. *The end of semester test (EOS) appears at the end of the course. Students are delivered a few items from every tutorial in the course in order to assess the major course objectives.*	1	20%
Total	**37**	**100%**

**Teachers may manually adjust these weights if desired, per district grading requirements.*

Scope and Sequence

This course is designed to prepare students to take the Advanced Placement (AP*) exams conducted by the College Board. The course is based on a comprehensive AP*syllabus that is rigorously aligned to the curricular design and test requirements of the College Board.

AP Course Audit Standards

UNIT 1: INTRODUCTION TO COMPUTER PROGRAMMING (DAYS 1 – 29)

In this unit, you will describe and explain the basic concepts of computer programming. You will compile and run a simple Java program. You will use unary and arithmetic operators, perform arithmetic calculations, and use the Math class. You will describe different types of errors, such as compile-time errors, runtime errors, and logical errors. You will also use relational and logical operators. You will identify the precedence of all Java operators. Finally, you will construct syntactically correct decision-making and loop statements, and implement algorithms.

Lesson/*Standards*	Lesson Objective
Syllabus and Orientation	Review the Student Orientation and Course Syllabus at the beginning of this course.
Computer Science	Describe and explain the basic concepts of computer programming.
Java Basics	Compile and run a simple Java program.
Variables and Data Types	Identify, describe, and employ Java variables and data types.
Arithmetic Operators and the Math Class	Describe and use unary and arithmetic operators, perform arithmetic calculations, and use the Math class.

Lesson/*Standards*	Lesson Objective
Relational and Logical Operators	Describe and use relational and logical operators.
Decision-Making Control Structures	Describe and use decision-making statements (if, if…else, nested if, switch).
Loops and Algorithms	Use different types of loop statements (while, for, nested) and implement algorithms.
Unit Activity: Introduction to Computer Programming	Generate random integer sets and perform mathematical operations, describe use of loops and decision-making statements to ensure flow control in the program, and suggest alternate ways of using loops and decision-making statements.

UNIT 2: CLASSES AND CONSTRUCTORS (DAYS 30 – 45)

In this unit, you will define a class and create objects for the class. You will differentiate between static and nonstatic fields. You will demonstrate how to achieve encapsulation in Java. You will define and call a constructor. You will declare a method and the parameters in that method. Finally, you will describe and employ different approaches to using parameters in methods.

Lesson/*Standards*	Lesson Objective
Classes	Describe, create, and properly use classes.
Methods	Describe, create, and call methods.
Method Parameters	Describe and employ different approaches to using parameters in methods.
Unit Activity: Classes and Constructors	Use and describe different types of methods to store student information; suggest alternate ways to use the class-related concepts; and suggest ways to make the program more robust.

UNIT 3: DATA STRUCTURES, SEARCHING, AND SORTING ALGORITHMS (DAYS 46 – 70)

In this unit, you will create the String object using string literal and new keyword. You will evaluate String expressions. You will create and manipulate one-dimensional and two-dimensional arrays. You will perform selection and insertion sort on an array and compare their performance. You will create a recursive method to solve a problem. You will implement merge sort. Finally, you will perform sequential and binary searches on an array and compare their performance.

Lesson/*Standards*	**Lesson Objective**
The String Class	Create String objects and use various String methods.
Arrays	Create and manipulate one-dimensional and two-dimensional arrays
The ArrayList Class	Create an ArrayList object and use various ArrayList methods.
Sorting Algorithms	Perform selection and insertion sort on an array and compare their performance.
Recursion and Merge Sort	Understand the concept of recursion and implement recursive methods to solve problems.
Searching Algorithms	Perform sequential and binary search on an array and compare their performance.
Unit Activity: Data Structures, Searching, and Sorting Algorithms	Use the concept of encapsulation in a class; use strings and arrays or array lists in a program; suggest alternate ways to use the class-related concepts and data structures; and suggest ways to make the program more robust.

UNIT 4: OBJECT-ORIENTED JAVA AND ETHICS IN COMPUTING (DAYS 71 – 90)

In this unit, you will explain abstraction and code reuse. You will demonstrate inheritance by extending a class. You will discuss the benefits of overriding. You will describe and implement polymorphism. You will create and extend an abstract class. You will create an interface. You will describe the ways to control viruses and other malicious attacks on computer systems. Finally, you will describe privacy rights, intellectual property rights, copyrights, fair use, piracy, and software license agreements.

Lesson/*Standards*	**Lesson Objective**
Object-Oriented Programming Design	Explain and implement object-oriented programming design.
Inheritance and Polymorphism	Explain the advantages of inheritance and polymorphism, and implement inheritance and polymorphism.
Abstraction and Interfaces	Describe and implement abstraction, and create an interface.
Ethics in Computing	Describe privacy and legality in the context of computing.
Unit Activity: Object-Oriented Java and Ethics in Computing	Use abstract classes and inheritance in a program; suggest alternate ways to use object-oriented programming concepts; and suggest ways to make the program more robust.

Course Map

You will achieve course level objectives by completing each lesson's instruction, assignments, and assessments. For a detailed look at how the materials meet these objectives, review the course map.