Unit 05: Writing Classes

Content Area: **Computer Science**
Course(s):
Time Period: **Marking Period 2**
Length: **3-4 Weeks**
Status: **Published**

Summary

This unit will pull together information from all previous units to create new, user-defined reference data types in the form of classes. The ability to accurately model real-world entities in a computer program is a large part of what makes computer science so powerful. This unit focuses on identifying appropriate behaviors and attributes of real-world entities and organizing these into classes.

Revision Date: July 2026

L.SS.11–12.1	Demonstrate command of the system and structure of the English language when writing or speaking.
9.4.12.CI.1	Demonstrate the ability to reflect, analyze, and use creative skills and ideas (e.g., 1.1.12prof.CR3a).
P.4	Demonstrate creativity and innovation.
P.5	Utilize critical thinking to make sense of problems and persevere in solving them.
8.1.12.CS.2	Model interactions between application software, system software, and hardware.
L.KL.11–12.2	Apply knowledge of language to understand how language functions in different contexts, to make effective choices for meaning or style, and to comprehend more fully when reading or listening.
8.1.12.CS.3	Compare the functions of application software, system software, and hardware.
P.8	Use technology to enhance productivity increase collaboration and communicate effectively.
P.9	Work productively in teams while using cultural/global competence.
8.1.12.AP.1	Design algorithms to solve computational problems using a combination of original and existing algorithms.
9.4.12.IML.3	Analyze data using tools and models to make valid and reliable claims, or to determine optimal design solutions (e.g., S-ID.B.6a., 8.1.12.DA.5, 7.1.IH.IPRET.8).
9.4.12.IML.4	Assess and critique the appropriateness and impact of existing data visualizations for an intended audience (e.g., S-ID.B.6b, HS-LS2-4).
8.1.12.AP.2	Create generalized computational solutions using collections instead of repeatedly using simple variables.
8.1.12.AP.5	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

Essential Questions & Essential Understanding

- How are access modifiers used to control the user interface with an object?
- How are objects created and used?
- How are tester classes effectively used?

- How do you access one object from another?
- What are reference variables?
- What are static variables and constants?
- What is a Class? What is an Object?
- What is object oriented programming (OOP)?
- What is the difference between zero and null?

Objectives

Students Will Know

- how to create object blueprints within a class by building the instance variables, constructors and methods.
- how to pass objects as references.
- the difference between an object and a primitive variable.

Students Will Be Skilled At

- encapsulating code to build useful but secure user interfaces.
- test object usage by building tester classes.
- recognizing the scope of a variable in an object, program and package.

Learning Plan

Discussion about what makes a phone a phone (or radio a radio, or pick an example). All phones have... some phones have... few phones have.... Classes include what all phones have - inheritance would add more but that isn't until a later unit. Blueprints to build any phone.

Lecture and demonstration of structure of a Class. Usually use a bank account as an example. Balance and account number are instance variables with the static variable next number used to teach static variables.

Group work developing classes and building tester classes to create objects and run all methods. Tester classes should also attempt to acces information that should be secure (all instance variabls are private) and test input arguments.

As a class, build the Student class based on feedback of what all students have (freshman - senior). Try to keep the amount of instance variables to name, grade, and one other variable. Accessors and mutator methods discussed. One year a major instance variable was stress, another year gpa, another sleep. All able to be accessed and adjusted. goToSchool and dayOff are methods covered that adjust info about the student.

Assessment

Assessments

- Formative: Daily assessments using examples from class notes and CodeHS.com, AP Classroom/Albert Checks for Understanding
- Summative: Teacher-created assessments/projects and CodeHS Computer Science Projects, AP Classroom/Albert Unit Assessments
- Benchmark: Check for understanding benchmark assessments on CodeHS, AP Classroom/Albert/Khan Academy Diagnostics
- Alternative Assessments: Student-centered activities such as a doorbell coding project, game design projects, and other activities involving real world applications

complete performance tasks:

- Students will be able to design programs using appropriate access modifiers, return types and structure of a class.
- Students will be able to write programs in order to solve given problems.

complete quizzes/tests:

- reference variables
- instance variables
- constructors
- modifiers and mutators
- scope of a variable
- static variables

complete sample AP multiple choice questions.

complete sample AP open ended questions.

Materials

Core instructional materials: Computing Concepts with Java

Supplemental instructional materials:

Java Concepts for AP Computer Science Study Guide
CollegeBoard AP Classroom Website
CollegeBoard AP Computer Science A Website

Suggested Strategies for Modifications

Possible accommodations/modification for AP Computer Science A