

AP COMPUTER SCIENCE A

UNIT 5

Writing Classes



5–7.5%
AP EXAM WEIGHTING



~12–14
CLASS PERIODS

The icon consists of a white circle containing a blue square with the letters 'AP' in white. Below the square is a small blue monitor-like shape with two vertical lines representing a stand.

Remember to go to [AP Classroom](#) to assign students the online **Personal Progress Check** for this unit.

Whether assigned as homework or completed in class, the **Personal Progress Check** provides each student with immediate feedback related to this unit's topics and skills.

Personal Progress Check 5

Multiple-choice: ~25 questions

Free-response: 2 questions

- Class
- Class: partial

Writing Classes



Developing Understanding

BIG IDEA 1

Modularity **MOD**

- How can using a model of traffic patterns improve travel time?

BIG IDEA 2

Variables **VAR**

- How can programs be written to protect your bank account balance from being inadvertently changed?

BIG IDEA 3

Impact of Computing **IOC**

- What responsibility do programmers have for the consequences of programs they create, whether intentional or not?

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. Students will build on what they learn in this unit to represent relationships between classes through hierarchies, which appear in Unit 9.

The creation of computer programs can have extensive impacts on societies, economies, and cultures. The legal and ethical concerns that come with programs and the responsibilities of programmers are also addressed in this unit.

Building Computational Thinking Practices

1.A 3.B 5.A 5.B 5.D

Computer scientists use computers to study and model the world, and this requires designing program code to fit a given scenario. Spending time to plan out the program up front will shorten overall program development time. Practicing elements of the iterative development process early and often will help create more robust programs and avoid logical errors.

Because revisions to program code are rarely completed by the original author, documenting program code is very important. It allows programmers to more quickly understand how a code segment functions without having to analyze the program code itself. Well-written documentation includes a description of the behavior, as well as the initial conditions that must be met for a code segment to work as intended.

Sometimes students struggle to explain why a code segment will not compile or work as intended. It helps to have students work with a collaborative partner to practice verbally

explaining the issues. The ability to explain why a code segment isn't working correctly assists in the development of solutions.

Preparing for the AP Exam

On the free-response section of the exam, students are required to design program code that demonstrates the functionality available for an object of a new class, based on the prompt's specification. This process includes identifying the attributes (data) that define an object of a class and the behaviors (methods) that define what an object of a class can do. Students should have many opportunities in this course to design their own classes based on program specifications and on observations of real-world objects.


Once the new class is designed, students should be able to implement program code to create a new type by creating a class. The behaviors of an object are expressed through writing *methods* in the class, which include expressions, conditional statements, and iterative statements. By being able to create their own classes, programmers are not limited to the existing classes provided within the Java libraries and can therefore represent their own ideas through classes.

UNIT AT A GLANCE

| Enduring Understanding | Topic | Suggested Skills | Class Periods |
|------------------------|---|---|----------------------|
| | | | ~12–14 CLASS PERIODS |
| MOD-2 MOD-3 | 5.1 Anatomy of a Class | <p>1.A Determine an appropriate program design to solve a problem or accomplish a task (<i>not assessed</i>).</p> <p>1.B Determine code that would be used to complete code segments.</p> | |
| | 5.2 Constructors | <p>1.C Determine code that would be used to interact with completed program code.</p> <p>3.B Write program code to define a new type by creating a class.</p> | |
| MOD-2 | 5.3 Documentation with Comments | 5.D Describe the initial conditions that must be met for a program segment to work as intended or described. | |
| | 5.4 Accessor Methods | <p>3.B Write program code to define a new type by creating a class.</p> <p>5.B Explain why a code segment will not compile or work as intended.</p> | |
| | 5.5 Mutator Methods | <p>3.B Write program code to define a new type by creating a class.</p> <p>4.B Identify errors in program code.</p> | |
| | 5.6 Writing Methods | <p>1.B Determine code that would be used to complete code segments.</p> <p>3.B Write program code to define a new type by creating a class.</p> | |
| | 5.7 Static Variables and Methods | <p>3.B Write program code to define a new type by creating a class.</p> <p>5.A Describe the behavior of a given segment of program code.</p> | |

continued on next page

UNIT AT A GLANCE *(cont'd)*

| Enduring Understanding | Topic | Suggested Skills | Class Periods |
|---|--|--|----------------------|
| | | | ~12–14 CLASS PERIODS |
| VAR-1 | 5.8 Scope and Access | 3.B Write program code to define a new type by creating a class. 5.B Explain why a code segment will not compile or work as intended. | |
| | 5.9 this Keyword | 2.C Determine the result or output based on the statement execution order in a code segment containing method calls. | |
| IOC-1 | 5.10 Ethical and Social Implications of Computing Systems | Curricular requirement, not assessed on the AP Exam | |
|  Go to AP Classroom to assign the Personal Progress Check for Unit 5. Review the results in class to identify and address any student misunderstandings. | | | |

SAMPLE INSTRUCTIONAL ACTIVITIES

The sample activities on this page are optional and are offered to provide possible ways to incorporate instructional approaches into the classroom. They were developed in partnership with teachers from the AP community to share ways that they approach teaching some of the topics in this unit. Please refer to the Instructional Approaches section beginning on p. 159 for more examples of activities and strategies.

| Activity | Topic | Sample Activity |
|----------|---------|---|
| 1 | 5.1 | <p>Kinesthetic learning</p> <p>Have students break into groups of 4–5 to play board games. Ask them to play the game for about 10 minutes. While they play the game, they should keep track of the various nouns they encounter and actions that happen as part of the game. The nouns can be represented in the computer as classes, and the actions are the behaviors. At the end of game play, ask students to create UML diagrams for the identified classes.</p> |
| 2 | 5.3 | <p>Marking the text</p> <p>Present students with specifications, and have them highlight or underline any preconditions (both implicit and explicit) that exist for the method to function. This includes information about parameters, such as object references not being null.</p> |
| 3 | 5.4–5.7 | <p>Create a plan</p> <p>When asked to write a method, have students write an outline using pseudocode with paper and pencil. Then, go through it step-by-step with sample input to ensure that the process is correct and to determine if any additional information is needed before beginning to program a solution on the computer.</p> |
| 4 | 5.7 | <p>Paraphrase</p> <p>Provide students with several example classes that utilize static variables for unique identification numbers or for counting the number of objects that have been created, but do not provide any description or documentation for the code. Have students spend time creating objects and calling the static methods to investigate how the static variables behave, then have them document the code appropriately to describe how each class utilizes static variables and methods.</p> |



Unit Planning Notes

Use the space below to plan your approach to the unit. Consider how you want to pace your course and where you will incorporate writing and analyzing program code.

.....

.....

.....

TOPIC 5.1

Anatomy of a Class

SUGGESTED SKILL

1.A

Determine an appropriate program design to solve a problem or accomplish a task.

1.B

Determine code that would be used to complete code segments.



AVAILABLE RESOURCES

- [Runestone Academy: AP CSA—Java Review: 2.7—Parts of a Java Class](#)
- [Practice-It!: BJP4 Chapter 8: Classes—Self-Check 8.1–8.7](#)
- [Classroom Resources > Object-Oriented Design](#)

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.A

Designate access and visibility constraints to classes, data, constructors, and methods.

ESSENTIAL KNOWLEDGE

MOD-2.A.1

The keywords `public` and `private` affect the access of classes, data, constructors, and methods.

MOD-2.A.2

The keyword `private` restricts access to the declaring class, while the keyword `public` allows access from classes outside the declaring class.

MOD-2.A.3

Classes are designated `public`.

MOD-2.A.4

Access to attributes should be kept internal to the class. Therefore, instance variables are designated as `private`.

MOD-2.A.5

Constructors are designated `public`.

MOD-2.A.6

Access to behaviors can be internal or external to the class. Therefore, methods can be designated as either `public` or `private`.

ENDURING UNDERSTANDING**MOD-3**

When multiple classes contain common attributes and behaviors, programmers create a new class containing the shared attributes and behaviors forming a hierarchy. Modifications made at the highest level of the hierarchy apply to the subclasses.

LEARNING OBJECTIVE**MOD-3.A**

Designate private visibility of instance variables to encapsulate the attributes of an object.

ESSENTIAL KNOWLEDGE**MOD-3.A.1**

Data encapsulation is a technique in which the implementation details of a class are kept hidden from the user.

MOD-3.A.2

When designing a class, programmers make decisions about what data to make accessible and modifiable from an external class. Data can be either accessible or modifiable, or it can be both or neither.

MOD-3.A.3

Instance variables are encapsulated by using the `private` access modifier.

MOD-3.A.4

The provided accessor and mutator methods in a class allow client code to use and modify data.

TOPIC 5.2

Constructors

SUGGESTED SKILLS

1.C

Determine code that would be used to interact with completed program code.

3.B

Write program code to define a new type by creating a class.



AVAILABLE RESOURCES

- [Runestone Academy: AP CSA—Java Review: 2.7—Parts of a Java Class](#)
- [Practice-It!: BJP4 Chapter 8: Classes—Exercises 8.14–8.22](#)

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.B

Define instance variables for the attributes to be initialized through the constructors of a class.

ESSENTIAL KNOWLEDGE

MOD-2.B.1

An object's state refers to its attributes and their values at a given time and is defined by instance variables belonging to the object. This creates a "has-a" relationship between the object and its instance variables.

MOD-2.B.2

Constructors are used to set the initial state of an object, which should include initial values for all instance variables.

MOD-2.B.3

Constructor parameters are local variables to the constructor and provide data to initialize instance variables.

MOD-2.B.4

When a mutable object is a constructor parameter, the instance variable should be initialized with a copy of the referenced object. In this way, the instance variable is not an alias of the original object, and methods are prevented from modifying the state of the original object.

MOD-2.B.5

When no constructor is written, Java provides a no-argument constructor, and the instance variables are set to default values.

SUGGESTED SKILL

5.D

Describe the initial conditions that must be met for a program segment to work as intended or described.



AVAILABLE RESOURCE

- External Resource > [Zereturnaround: Reasons, Tips and Tricks for Better Java Documentation](#)

TOPIC 5.3

Documentation with Comments

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.C

Describe the functionality and use of program code through comments.

ESSENTIAL KNOWLEDGE

MOD-2.C.1

Comments are ignored by the compiler and are not executed when the program is run.

MOD-2.C.2

Three types of comments in Java include `/* */`, which generates a block of comments, `//`, which generates a comment on one line, and `/** */`, which are Javadoc comments and are used to create API documentation.

MOD-2.C.3

A precondition is a condition that must be true just prior to the execution of a section of program code in order for the method to behave as expected. There is no expectation that the method will check to ensure preconditions are satisfied.

MOD-2.C.4

A postcondition is a condition that must always be true after the execution of a section of program code. Postconditions describe the outcome of the execution in terms of what is being returned or the state of an object.

MOD-2.C.5

Programmers write method code to satisfy the postconditions when preconditions are met.

TOPIC 5.4

Accessor Methods

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.D

Define behaviors of an object through non-void methods without parameters written in a class.

ESSENTIAL KNOWLEDGE

MOD-2.D.1

An accessor method allows other objects to obtain the value of instance variables or static variables.

MOD-2.D.2

A non-void method returns a single value. Its header includes the return type in place of the keyword `void`.

MOD-2.D.3

In non-void methods, a return expression compatible with the return type is evaluated, and a copy of that value is returned. This is referred to as “return by value.”

MOD-2.D.4

When the return expression is a reference to an object, a copy of that reference is returned, not a copy of the object.

MOD-2.D.5

The `return` keyword is used to return the flow of control to the point immediately following where the method or constructor was called.

SUGGESTED SKILLS

3.B

Write program code to define a new type by creating a class.

5.B

Explain why a code segment will not compile or work as intended.



AVAILABLE LABS

- Classroom Resources >
 - AP Computer Science A: Data Lab
 - AP Computer Science A: Celebrity Lab

AVAILABLE RESOURCE

- Practice-It!: BJP4 Chapter 8: Classes—Exercises 8.14–8.22

continued on next page

LEARNING OBJECTIVE**MOD-2.D**

Define behaviors of an object through non-void methods without parameters written in a class.

ESSENTIAL KNOWLEDGE**MOD-2.D.6**

The `toString` method is an overridden method that is included in classes to provide a description of a specific object. It generally includes what values are stored in the instance data of the object.

MOD-2.D.7

If `System.out.print` or `System.out.println` is passed an object, that object's `toString` method is called, and the returned string is printed.

TOPIC 5.5

Mutator Methods

SUGGESTED SKILLS**3.B**

Write program code to define a new type by creating a class.

4.B

Identify errors in program code.

**AVAILABLE RESOURCE**

- Practice-It! BJP4 Chapter 8: Classes—Exercises 8.14–8.22

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.E

Define behaviors of an object through void methods with or without parameters written in a class.

ESSENTIAL KNOWLEDGE

MOD-2.E.1

A void method does not return a value. Its header contains the keyword `void` before the method name.

MOD-2.E.2

A mutator (modifier) method is often a void method that changes the values of instance variables or static variables.

SUGGESTED SKILLS

1.B

Determine code that would be used to complete code segments.

3.B

Write program code to define a new type by creating a class.



AVAILABLE RESOURCES

- Practice-It!: BJP4
Chapter 3: Parameters and Objects—
Exercises 3.1–3.22
- Practice-It!: BJP4
Chapter 8: Class—
Exercises 8.14–8.22

TOPIC 5.6

Writing Methods

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.F

Define behaviors of an object through non-void methods with parameters written in a class.

ESSENTIAL KNOWLEDGE

MOD-2.F.1

Methods can only access the private data and methods of a parameter that is a reference to an object when the parameter is the same type as the method's enclosing class.

MOD-2.F.2

Non-void methods with parameters receive values through parameters, use those values, and return a computed value of the specified type.

MOD-2.F.3

It is good programming practice to not modify mutable objects that are passed as parameters unless required in the specification.

MOD-2.F.4

When an actual parameter is a primitive value, the formal parameter is initialized with a copy of that value. Changes to the formal parameter have no effect on the corresponding actual parameter.

continued on next page

LEARNING OBJECTIVE

MOD-2.F

Define behaviors of an object through non-void methods with parameters written in a class.

ESSENTIAL KNOWLEDGE

MOD-2.F.5

When an actual parameter is a reference to an object, the formal parameter is initialized with a copy of that reference, not a copy of the object. If the reference is to a mutable object, the method or constructor can use this reference to alter the state of the object.

MOD-2.F.6

Passing a reference parameter results in the formal parameter and the actual parameter being aliases. They both refer to the same object.

SUGGESTED SKILLS

3.B

Write program code to define a new type by creating a class.

5.A

Describe the behavior of a given segment of program code.

TOPIC 5.7

Static Variables and Methods

Required Course Content

ENDURING UNDERSTANDING

MOD-2

Programmers use code to represent a physical object or nonphysical concept, real or imagined, by defining a class based on the attributes and/or behaviors of the object or concept.

LEARNING OBJECTIVE

MOD-2.G

Define behaviors of a class through static methods.

ESSENTIAL KNOWLEDGE

MOD-2.G.1

Static methods are associated with the class, not objects of the class.

MOD-2.G.2

Static methods include the keyword `static` in the header before the method name.

MOD-2.G.3

Static methods cannot access or change the values of instance variables.

MOD-2.G.4

Static methods can access or change the values of static variables.

MOD-2.G.5

Static methods do not have a `this` reference and are unable to use the class's instance variables or call non-static methods.

MOD-2.H

Define the static variables that belong to the class.

MOD-2.H.1

Static variables belong to the class, with all objects of a class sharing a single static variable.

continued on next page

LEARNING OBJECTIVE

MOD-2.H

Define the static variables that belong to the class.

ESSENTIAL KNOWLEDGE

MOD-2.H.2

Static variables can be designated as either `public` or `private` and are designated with the `static` keyword before the variable type.

MOD-2.H.3

Static variables are used with the class name and the dot operator, since they are associated with a class, not objects of a class.

SUGGESTED SKILLS

5.B

Explain why a code segment will not compile or work as intended.

3.B

Write program code to define a new type by creating a class.

TOPIC 5.8

Scope and Access

Required Course Content

ENDURING UNDERSTANDING

VAR-1

To find specific solutions to generalizable problems, programmers include variables in their code so that the same algorithm runs using different input values.

LEARNING OBJECTIVE

VAR-1.G

Explain where variables can be used in the program code.

ESSENTIAL KNOWLEDGE

VAR-1.G.1

Local variables can be declared in the body of constructors and methods. These variables may only be used within the constructor or method and cannot be declared to be `public` or `private`.

VAR-1.G.2

When there is a local variable with the same name as an instance variable, the variable name will refer to the local variable instead of the instance variable.

VAR-1.G.3

Formal parameters and variables declared in a method or constructor can only be used within that method or constructor.

VAR-1.G.4

Through method decomposition, a programmer breaks down a large problem into smaller subproblems by creating methods to solve each individual subproblem.

TOPIC 5.9

this Keyword

SUGGESTED SKILL**2.C**

Determine the result or output based on the statement execution order in a code segment containing method calls.

**AVAILABLE RESOURCE**

- Past AP Free-Response Exam Questions on Class on AP Question Bank

Required Course Content

ENDURING UNDERSTANDING

VAR-1

To find specific solutions to generalizable problems, programmers include variables in their code so that the same algorithm runs using different input values.

LEARNING OBJECTIVE

VAR-1.H

Evaluate object reference expressions that use the keyword `this`.

ESSENTIAL KNOWLEDGE

VAR-1.H.1

Within a non-static method or a constructor, the keyword `this` is a reference to the current object—the object whose method or constructor is being called.

VAR-1.H.2

The keyword `this` can be used to pass the current object as an actual parameter in a method call.



AVAILABLE RESOURCE

- Classroom Resources >
[Ethical Use of the Computer](#)

TOPIC 5.10

Ethical and Social Implications of Computing Systems

Required Course Content

ENDURING UNDERSTANDING

IOC-1

While programs are typically designed to achieve a specific purpose, they may have unintended consequences.

LEARNING OBJECTIVE

IOC-1.A

Explain the ethical and social implications of computing systems.

ESSENTIAL KNOWLEDGE

IOC-1.A.1

System reliability is limited. Programmers should make an effort to maximize system reliability.

IOC-1.A.2

Legal issues and intellectual property concerns arise when creating programs.

IOC-1.A.3

The creation of programs has impacts on society, economies, and culture. These impacts can be beneficial and/or harmful.