Syllabus Development Guide: AP Computer Science AB

<u>To the AP teacher</u>: Please take full advantage of this guide. It is designed to support you as you develop your syllabus for the AP Course Audit. The guide contains the following sections and information:

Curricular Requirements — The curricular requirements are the core elements of the course. Your syllabus must provide clear evidence that each requirement is fully addressed in your course.

Scoring Components — Some curricular requirements consist of complex, multi-part statements. These particular requirements are broken down into their component parts and restated as "scoring components". Reviewers will look for evidence that each scoring component is included in your course.

Key Terms — To ensure the clarity of certain terms or expressions that may have multiple meanings, each of these terms is clearly defined.

Evaluation Guidelines — These are the exact guidelines used by reviewers as they evaluate the evidence in your syllabus. Use these to interpret any requirement you may find ambiguous.

Important Considerations — Aligned with the Evaluation Guidelines, these statements provide advice on the type of evidence your syllabus should include.

Reference — As appropriate, references to specific sections of the official AP Course Description or other pertinent publications are included here.

Samples of Evidence — For each scoring component, three separate samples of evidence are provided. These statements provide either verbatim samples from actual authorized syllabi or clear descriptions of what acceptable evidence should look like.

<table>
<tr><th>Curricular Requirements</th><th colspan="4">Scoring Components, Key Terms, Evaluation Guidelines, Important Considerations, References and Samples of Evidence</th></tr>
<tr><td rowspan="6">Curricular Requirement 1: The course includes all of the topics listed in the "Computer Science AB" columns of the Topic Outline in the AP Computer Science Course Description.</td><td colspan="4">Scoring Component 1*: The course teaches students object-oriented program design principles.</td></tr>
<tr><td colspan="4">*Note Each Curricular Requirement may be subdivided into two or more distinct Scoring Components.</td></tr>
<tr><td>Key Term(s)</td><td>Evaluation Guideline(s)</td><td>Important Consideration(s)</td><td>Reference</td></tr>
<tr><td>Object-oriented program design principles: learn how to design a hierarchy of interacting classes using "is-a" and "has-a" relationships; apply concepts of encapsulation, inheritance and polymorphism to design solutions to problems; learn to model objects within a problem domain with classes; and learn how to use given, existing classes together with new classes.</td><td>The syllabus must contain all of the following topics: classes; methods and parameters; encapsulation; inheritance and composition; and interfaces.

The syllabus must include an assignment/project that involves designing and implementing a solution using a set of classes that are related by inheritance and/or composition.

If the syllabus contains a unit on Object-Oriented design with UML, then this scoring component is met.</td><td>At least one of the assignments/projects/labs should be detailed enough to demonstrate how multiple related classes were used in the solution.

“is-a” and “has-a” relationships among classes should appear in the syllabus’ course outline or schedule.

Data-hiding or equivalent terms can replace 'encapsulation.'

Inheritance and composition can replace “is-a” and “has-a” relationships.</td><td>For additional information see “I. Object-Oriented Program Design” on pages 12 – 14 of the AP Computer Science Course Description.</td></tr>
<tr><td colspan="4">Samples of Evidence</td></tr>
<tr><td>Sample 1</td><td>Sample 2</td><td colspan="2">Sample 3</td></tr>
<tr><td>The syllabus includes units focusing on the following topics: classes, inheritance, abstract classes, encapsulation and interfaces, and one of the units requires students to design an application with interacting classes and subclasses.</td><td>The syllabus mentions the creation of a hierarchy of mathematical shapes that includes:
• accessor and modifier methods
• use of a ListIterator to iterate through the shape objects to display information.
• calculation of the area of a complex geometrical shape that is made up of two or more simple shapes.</td><td colspan="2">The syllabus mentions designing and implementing a banking application. The design includes a BankAccount class with subclasses CheckingAccount and SavingsAccount; a BankTransaction class that implements a Transaction interface and “has a” Date, the BankAccount information, the amount of the transaction, and the transaction type.</td></tr>
</table>

Curricular Requirement 2: The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.	**Scoring Component 2: The course teaches students to design and implement computer-based solutions to problems.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	Design: defining the class(es) with their respective attributes, constants and operations, the interactions among classes, and the algorithms that are required to solve a specific problem. **Implement:** creating those classes by writing the Java code and running and testing the solution designed.	The syllabus must include explicit evidence that students engage in regular, frequent practice writing programs. Programming activity (assignments/labs/projects) must be included in most units, roughly 75% of the time. Representative examples of assignment titles and/or short descriptions must be included in the syllabus. A list of textbook problem numbers alone is not sufficient evidence. For example, "Various assignments from Chapter 3" is not sufficient evidence; likewise, "Problems 8.2, 8.3, 8.4" is not sufficient evidence.	The syllabus should clearly demonstrate that students design solutions to problems using object-oriented constructs, data structures and algorithms; that students implement, document and test those solutions; and that students learn how to solve problems by creating working software. Evidence of design could include exercises in writing, documenting (for example, by using Javadoc, commenting, pre/post conditions, and/or assertions), and testing programs. Evidence of implementation could include exercises and problems that involve coding with object-oriented constructs, data structures, and algorithms.	For additional information see "I. Object-Oriented Program Design" and "II. Program Implementation" on pages 12 – 18 of the AP Computer Science Course Description.
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus includes a variety of problems involving interacting classes under each unit of study, including Lottery, Slot Machine, Pac Fish, Asteroids, Spell Checker, and Google Billboard.	The syllabus includes a variety of projects involving interacting classes integrated throughout the course including a project where students design and implement an application for a registrar's office. This application has classes Student, Course, CourseSection and Instructor.	The syllabus includes units focusing on the following topics: classes, inheritance, abstract classes, encapsulation and interfaces, and one of the units requires students to design an application with interacting classes and subclasses that includes the classes Student, HighSchoolStudent, CollegeStudent, UndergraduateStudent, GraduateStudent.	

<table>
<tr><td rowspan="6">Curricular Requirement 2 (continued): The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.</td><td colspan="4">Scoring Component 3: Computer-based programs in the course include a variety of application areas.</td></tr>
<tr><td>Key Term(s)</td><td>Evaluation Guideline(s)</td><td>Important Consideration(s)</td><td>Reference</td></tr>
<tr><td>A variety of application areas: at least two different domain areas. For a representative sample of assignments, the domain areas should be explicitly mentioned using a sentence or two.</td><td>At least two application areas must be present in the syllabus.</td><td>The programs students design, implement and test should be drawn from a variety of application domains, which may include, but are not limited to financial, gaming, natural and social sciences, engineering or math applications.</td><td>No references to external documents are needed for this Scoring Component.</td></tr>
<tr><td colspan="4">Samples of Evidence</td></tr>
<tr><td>Sample 1</td><td>Sample 2</td><td colspan="2">Sample 3</td></tr>
<tr><td>The syllabus mentions designing and implementing tic-tac-toe and simulation of a hospital emergency room using priority queues.</td><td>The syllabus mentions implementation of traversing a maze using two-dimensional arrays and recursion and a library simulation using HashSet.</td><td colspan="2">The syllabus mentions the implementation of an E-mail directory using HashSet and RPN using stacks.</td></tr>
</table>

Curricular Requirement 3: The course teaches students to use and implement commonly used algorithms and data structures.	**Scoring Component 4: The course teaches students to use and implement commonly-used algorithms.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	Use: knowing how to integrate classes and methods written by someone else (the Java implementers, the teacher, or the textbook author, for example). **Implement:** knowing how to write and test Java classes and methods "from scratch" - translating algorithms described without Java (In English or mathematics, for example) into working Java code.	The syllabus must provide evidence that students work with sorting and searching algorithms. The syllabus must explicitly mention sequential and binary search; insertion, selection, merge, heap and quick sorts. The syllabus must include programming activity with insertion to, deletions from, and traversals of various data structures.	The syllabus should include some activity involving algorithms either by implementing a program or by comparing running times of algorithms on various inputs.	For additional information see page 10 and “V. Standard Algorithms” on page 25 of the AP Computer Science Course Description.
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus includes the following assignments: o Write methods to insert to, delete from, and traverse items in lists, binary trees, heaps, queues, stacks, sets, and maps. o Trace each of the 5 sorting algorithms listed in the Course Description through at least one pass or recursive call. o Compare the number of comparisons made in searching for a particular element using a linear search and a binary search.	The syllabus discusses analysis of algorithms and explains how Big-Oh is determined for each the five sorting algorithms, the linear search, the binary search, insertion to, deletion from, and traversal of different data structures.	The syllabus provides evidence of instruction for O(n2) ,O(n log n) , O(n) and O(log n) algorithms by analyzing o the five sort algorithms listed in the Course Description. o the two search algorithms listed in the Course Description. o insertions, deletions, and traversals of various data structures.	

Curricular Requirement 3 (continued): The course teaches students to use and implement commonly used algorithms and data structures.	**Scoring Component 5: The course teaches students to use and implement commonly used-data structures.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	Use and implement: creating and manipulating arrays, stacks, queues, lists, priority queues, hash tables, trees, sets and maps.	The syllabus must contain the following topics: arrays, stacks, queues, lists, priority queues, hash tables, trees, sets and maps. The syllabus must include programming activity with various data structures. The syllabus must include at least one programming assignment using stacks, queues, or priority queues; one programming assignment using lists; and one programming assignment using sets or maps.	The syllabus should include at least one assignment that uses one of the following data structures: stacks, queues, trees, priority queues, lists, sets, and maps.	For additional information, see page 10 and "IV. Standard Data Structures" on page 21 – 24 of the AP Computer Science Course Description.
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus includes the following problem statements or lecture topics: • Use a HashMap or TreeMap to store countries and their capitals. • Discuss the differences between stacks and queues. • Discuss the differences between queues and priority queues. • Discuss the relationships between heaps and priority queues. • Discuss the differences of ArrayLists and linked lists. • Implement a BinarySearchTree class.	The syllabus • provides evidence of instruction for O(n2) ,O(n log n) , O(n) and O(log n) analysis of insertion to, deletion from : stacks, queues, binary search trees, arrays, lists, maps (tree and hash), sets(tree and hash) and priority queues. • requires writing an implementation for a queue class that implements the Queue interface. • requires writing a client program that demonstrates the use of the java.util.LinkedList methods included in the AP CS subset.	The syllabus provides evidence of instruction for O(n2) ,O(n log n) , O(n) and O(log n) algorithms by analyzing • the five sort algorithms listed in the Course Description. • the two search algorithms listed in the Course Description. • insertions, deletions, and traversals of various data structures.	

<table>
<tr><td rowspan="6">Curricular Requirement 4: The course teaches students to develop and select appropriate algorithms and data structures to solve problems.</td><td colspan="4">Scoring Component 6: The course teaches students to select appropriate algorithms and data structures to solve problems.</td></tr>
<tr><td>Key Term(s)</td><td>Evaluation Guideline(s)</td><td>Important Consideration(s)</td><td>Reference</td></tr>
<tr><td>Select appropriate algorithms and data structures: determining which among several alternative algorithms or data structures is more appropriate for specific applications.</td><td>The syllabus must provide evidence of instruction of comparison of algorithms and data structures to solve problems.

“Big-Oh” must be explicitly mentioned in the syllabus.</td><td>The syllabus should include evidence of instruction and exercises in the following:
• Time complexity of operations on data structures
• How to select a particular data structure and/or algorithm based on efficiency of operations.
• Which data structures are appropriate for an application
• Big -Oh and how to express complexity of algorithms and operations on data structures using Big -Oh.
• Worst and average case scenarios</td><td>For additional information see “III. Program Analysis” on page 9 and 19 – 21 of the AP Computer Science Course Description.</td></tr>
<tr><td colspan="4">Samples of Evidence</td></tr>
<tr><td>Sample 1</td><td>Sample 2</td><td colspan="2">Sample 3</td></tr>
<tr><td>The syllabus includes a class lecture on the runtime and memory tradeoffs of array based and linked list based implementations of linear collections and includes a lecture topic on the behavior of a queue and identifies situations where a queue is the appropriate data structure to use.</td><td>The syllabus includes a class discussion on why a hashing implementation of sets and maps can be very efficient.</td><td colspan="2">The syllabus states that students will
• understand the expected running time of the HashSet and TreeSet implementations of the Set interface.
• understand the expected running time of the HashMap and TreeMap implementations of the Map interface</td></tr>
</table>

Curricular Requirement 5: The course teaches students to code fluently in an object-oriented paradigm using the programming language Java. The course teaches students to use standard Java library classes from the AP Java subset delineated in Appendixes A and C of the AP Computer Science Course Description. (Note: students who study a language other than Java during an AP Computer Science course must also be taught to use Java, as specified in the AP Java subset.)	**Scoring Component 7: The course teaches students to code fluently in an object-oriented paradigm using the programming language Java.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	All terminology in the Scoring Component is clear. No clarification is needed.	The syllabus must state the use of Java as the programming language. If the syllabus includes evidence of implementing classes involving the GridWorld Case Study, then this scoring component is met. If scoring component 1 is not met, then this scoring component is not met. Students who study a language other than Java during an AP Computer Science course must also be taught to use Java, as specified in the AP Java subset.	Scoring Component is clear and explicit. No Important Considerations are needed.	No references to external documents are needed for this Scoring Component.
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus includes exercises from the GridWorld Case Study extending the Bug and Critter classes.	The syllabus includes the following exercise: Extend a given BankAccount class by writing a SavingsAccount class. The BankAccount class provides methods for inquiring about the balance in the account, making deposits, and making withdrawals. The SavingsAccount class also allows inquires, deposits, and withdrawals but, unlike the general kind of account, updates the balance during each operation by computing accumulated interest.	The syllabus mentions the use of classes belonging to the java.util and java.lang packages and has references to the AP Java subset throughout the course.	

<table>
<tr><td rowspan="6">Curricular Requirement 5 (continued): The course teaches students to code fluently in an object-oriented paradigm using the programming language Java. The course teaches students to use standard Java library classes from the AP Java subset delineated in Appendixes A and C of the AP Computer Science Course Description. (Note: students who study a language other than Java during an AP Computer Science course must also be taught to use Java, as specified in the AP Java subset.)</td><td colspan="4">Scoring Component 8: The course teaches students to use standard Java library classes from the AP Java subset delineated in Appendix C of the AP Computer Science Course Description.</td></tr>
<tr><td>Key Term(s)</td><td>Evaluation Guideline(s)</td><td>Important Consideration(s)</td><td>Reference</td></tr>
<tr><td>All terminology in the Scoring Component is clear. No clarification is needed.</td><td>The syllabus must provide evidence of instruction of standard Java library classes, methods, and interfaces from the AP Java subset.

If the syllabus mentions the use of the AP Computer Science Quick Reference Guide in the course then this scoring component is met.

If the syllabus mentions the use of the Appendix C of the AP Computer Science Course Description in the course then this scoring component is met.</td><td>Evidence may be inferred that the course introduces students to the standard Java library classes, methods, and interfaces from the AP Java subset through examples, e.g., lectures, study topics, labs, assignments and exercises in the syllabus (see examples below).</td><td>For additional information see page 23 – 24 and Appendices A and C in the AP Computer Science Course Description.</td></tr>
<tr><td colspan="4">Samples of Evidence</td></tr>
<tr><td>Sample 1</td><td>Sample 2</td><td colspan="2">Sample 3</td></tr>
<tr><td>The syllabus states that the students use the AP CS Quick Reference Guide during all quizzes and exams and as a reference for all assignments.</td><td>The syllabus includes
o assignments using at least two of the Collection classes such as java.util.LinkedList and java.util.HashSet.
o assignments using the String, Math, and wrapper classes.</td><td colspan="2">The syllabus includes
• using LinkedList class and ListIterator interface to program a mergelist algorithm
• implementing heapsort using PriorityQueue class
• implementing Queue interface with a LinkedList
• creating a Grid class using a Map, Set, and a sparse matrix and comparing their performances.</td></tr>
</table>

Curricular Requirement 6: The course teaches students to read and understand a large program consisting of several classes and interacting objects, and enables students to read and understand the current AP Computer Science Case Study posted on AP Central.	**Scoring Component 9: The course teaches students to read and understand a large program consisting of several classes and interacting objects. In particular, the course enables students to read and understand the *current AP Computer Science Case Study*, posted on AP Central.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	Current AP Computer Science Case Study: GridWorld Case Study	The syllabus must include all five parts of the GridWorld Case Study.	The GridWorld Case Study can be integrated throughout the course either as part of instruction on data structures or by including assignments from the case study at various stages of the course. It is not necessary that an entire unit be spent on the GridWorld Case Study.	For additional information see "Case Studies" on page 27 of the AP Computer Science Course Description. In addition, the Gridworld Case Study can be downloaded from AP central at http://apcentral.collegeboard.com/apc/public/courses/teachers_corner/8153.html
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus includes a unit on GridWorld Case Study with the following activities: • Run, analyze and experiment with the GridWorld Case Study. • Understand the Bug class, Runner Class and Grid Interface. • Extend the Bug Class by creating a specialized bug to meet requested requirements. • Use Inheritance to extend the Critter class by making new types of critters • Contrast between various implementations of BoundedGrid class	The syllabus integrates the GridWorld Case study throughout the course such as: • When introducing inheritance, problems extending the Bug class are assigned. • When introducing interacting classes, discussions and problems involving the Critter, the ChameleonCritter, and other subclasses of Critter are included. • When discussing 2-dimensional arrays, the code for the BoundedGrid class is discussed. • When maps are introduced, analysis comparison discussion abound 2-D array implementation and map implementation of the BoundedGrid are included.	The syllabus includes take-home free response questions for each of the five parts of the GridWorld Case Study and also assigns a problem set of 30 Multiple-Choice GridWorld questions.	

Curricular Requirement 7: The course teaches students to identify the major hardware and software components of a computer system, their relationship to one another, and the roles of these components within the system.	**Scoring Component 10: The course teaches students to identify the major hardware and software components of a computer system, their relationship to one another, and the roles of these components within the system.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	Major hardware and software components; **relationships;** and **roles:** collectively interpreted as acquainting students with the tools they will use to create software.	The syllabus must include instruction in at least one of the following: • Hardware and software components; • Their relationships to each other; or • The roles of these components within the system If the syllabus states that the topic was addressed in another course, then it must also include content and materials taught in that course (i.e., a text book citation and chapter titles or a descriptive narrative).	This topic is often integrated into numerous sections in a course; it is not necessary for the syllabus to describe instruction of hardware or software in great detail.	For additional information see page 11 and "VI. Computing in Context" on pages 25 – 26 of the AP Computer Science Course Description.
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus includes a lecture on software and hardware.	The syllabus includes descriptive narrative about a class discussion on how the computer lab is set up and it's various components and their functionalities.	The syllabus includes an activity where students dissemble and rebuild a computer in class.	

Curricular Requirement 8: The course teaches students to recognize the ethical and social implications of computer use.	**Scoring Component 11: The course teaches students to recognize the ethical and social implications of computer use.**			
	Key Term(s)	Evaluation Guideline(s)	Important Consideration(s)	Reference
	Recognize the ethical and social implications: any activity that engages students in discussions of principles for the responsible use of computers.	The syllabus must include explicit evidence of how the ethical or social implications of computer use are addressed within the course. If the syllabus does not identify an activity (paper, presentation, etc.), then it must at least identify topics of discussion in the course schedule (protection of privacy, intellectual property, public safety, etc.). If the syllabus states that this topic was addressed in another course, then it must also include content and materials taught in that course (e.g., a textbook citation and chapter titles or a descriptive narrative). Optional or extra credit assignments on ethical and social implications of computer use are not sufficient evidence. Reference to an "acceptable use of school computer" policy alone is not sufficient evidence. Course goals and/or objectives alone are not sufficient evidence.	Evidence that the course teaches students to recognize the ethical and social implications of computer use can be illustrated in a variety of ways, including but not limited scheduled lessons or discussion of "ethical use of technology," "responsible use of technology," or discussions of one or more of topics like the following: • Respect for/protection of privacy • Respect for/protection of intellectual property • Defense against vandalism, fraud, and other kinds of crime on the Internet • Potential of new technologies to affect public safety • Potential of new technologies to affect employment • Liability of software engineers for defects in their products • Inclusion, free speech, Section 508 compliance (US law that requires equitable access)	For additional information see "D. Responsible use of computer systems" under "VI. Computing in Context" on page 11 and "VI. Computing in Context" on pages 25 – 26 of the AP Computer Science Course Description.
	Samples of Evidence			
	Sample 1	Sample 2	Sample 3	
	The syllabus states that students engage in social, ethical, and system reliability journaling throughout the course and includes sample prompts given to students.	The syllabus includes a required writing assignment in which students research and discuss a specific topic related to ethical implications of computer use.	The syllabus includes the following topics in the course schedule under "Unit 1": copyright laws, software piracy, intellectual property, privacy, and network reliability.	