CollegeBoard AP

SAMPLE SYLLABUS #1

AP® Computer Science A

Curricular Requirements

CR1	Students and teachers have access to a college-level computer science textbook in print or electronic format	*See page:* 2
CR2	The course provides opportunities to develop student understanding of the required content outlined in each of the units described in the AP Course and Exam Description (CED).	*See page:* 2
CR3	The course provides opportunities to develop student understanding of the big ideas.	*See page:* 8
CR4	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 1: Program Design and Algorithm Development.	*See page:* 8
CR5	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 2: Code Logic.	*See page:* 8
CR6	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 3: Code Implementation.	*See page:* 8
CR7	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 4: Code Testing.	*See page:* 8
CR8	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 5: Documentation	*See page:* 9
CR9	This course provides students with hands-on lab experiences to practice programming through designing and implementing computer-based solutions to problems.	*See pages:* 2, 8

Advanced Placement Computer Science A Sample Syllabus #1

Overview

Students in this class will exceed the 20 hour in-class programming requirement, probably before the end of the first quarter. CR9 In addition to writing dozens of programs throughout the year, students will also complete a larger programming project at the end of each semester.

CR9
The syllabus must include an explicit statement that at least 20 hours of in-class instructional time is spent in computer-based lab experiences.

Texts and Resources

The following texts are used in the course:

- *Java Concepts: AP® Edition* (JC), Cay Horstmann, 5th Edition, 2008, John Wiley & Sons, Inc. CR1
- *AP Computer Science Study Guide* (APSG), Frances P. Trees, 4th Edition, 2006, John Wiley & Sons, Inc.
- *Introduction to Computing & Programming with JAVA: A Multimedia Approach* (MM), Guzdial & Ericson, 2007, Pearson Education, Inc.

CR1
The syllabus must list the title and author of a college-level computer science textbook.

The classroom has laptops for each student and internet access. In addition, ours is a bring-your-own-device school, and most students have laptops at least equal to the school laptops.

Student Practice

Throughout each unit, **Topic Questions** will be provided to help students check their understanding. The Topic Questions are especially useful for confirming understanding of difficult or foundational topics before moving on to new content or skills that build upon prior topics. Topic Questions can be assigned before, during, or after a lesson, and as in-class work or homework. Students will get rationales for each Topic Question that will help them understand why an answer is correct or incorrect, and their results will reveal misunderstandings to help them target the content and skills needed for additional practice.

At the end of each unit or at key points within a unit, **Personal Progress Checks** will be provided in class or as homework assignments in AP Classroom. Students will get a personal report with feedback on every topic, skill, and question that they can use to chart their progress, and their results will come with rationales that explain every question's answer. One to two class periods are set aside to re-teach skills based on the results of the Personal Progress Checks.

Course Outline

CR2 In this outline, the *CED Unit* designation matches each of my units with a unit from the 2019–2020 course description. I have assignments to discuss general ethical concerns in computing and the responsibilities of programmers specifically. **Assignment** marks a programming or written assignment; those with asterisks before the names are described in a separate section after the outline.

CR2
The syllabus must include an outline of course content by unit title using any organizational approach to demonstrate the inclusion of required course content.

Unit 1: Primitive Types		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
1.1 Why Programming? Why Java?	2.B, 4.B	
1.2 Variables and Data Types	1.A, 1.B	
1.3 Expressions and Assignment Statements	1.B, 2.A	
1.4 Compound Assignment Operators	2.B, 5.A	Assignment: Dollars and Cents Assignment: Paper to IDE
1.5 Casting and Ranges of Variables	2.B, 5.B	
Complete Personal Progress Checks for Unit 1		Personal Progress Check MCQ Part A Personal Progress Check MCQ Part B
Unit 1 Review		
Unit 1 Test		

Unit 2: Using Objects		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
2.1 Objects–Instances of Classes	5.A	
2.2 Creating and Storing Objects (Instantiation)	1.C, 3.A	
2.3 Calling a Void Method	1.C, 3.A	
2.4 Calling a Void Method with Parameters	2.C, 3.A	
2.5 Calling a Non-void Method	1.C, 3.A	
2.6 `String` Objects: Concatenation, Literals, and More	2.A	
2.7 `String` Methods	2.C, 3.A	Lab 1: Splitting Strings
2.8 Wrapper Classes: `Integer` and `Double`	2.C	
2.9 Using the `Math` Class	1.B, 3.A	
Complete Personal Progress Checks for Unit 2		Personal Progress Check MCQ Part A Personal Progress Check MCQ Part B Personal Progress Check FRQ
Unit 2 Review		
Unit 2 Test		

Unit 3: Boolean Expressions and `if` Statements		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
3.1 Boolean Expressions	2.A	Assignment: *Output or Trick
3.2 `if` Statements and Control Flow	2.B, 3.C	
3.3 `if-else` Statements	3.C, 4.A	
3.4 `else if` Statements	3.C, 4.C	
3.5 Compound Boolean Expressions	2.B, 3.C	
3.6 Equivalent Boolean Expressions	4.C	
3.7 Comparing Objects	2.C, 3.A	
Complete Personal Progress Checks for Unit 3		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 3 Review		
Unit 3 Test		

Unit 4: Iteration		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
4.1 `while` Loops	1.B, 2.B, 3.C	Lab 2: *Rolling Dice
4.2 `for` Loops	3.C, 4.C, 5.C	Assignment: Matching Positions Assignment: For-loop Patterns
4.3 Developing Algorithms Using Strings	2.C, 3.C	
4.4 Nested Iteration	1.B, 3.C, 5.C	Lab 3: Processing from a File
4.5 Informal Code Analysis	2.D	
Complete Personal Progress Checks for Unit 4		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 4 Review		
Unit 4 Test		

Unit 5: Writing Classes		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
5.1 Anatomy of a Class	1.A, 1.B	Lab 4: Investments and Investors
5.2 Constructors	1.C, 3.B	
5.3 Documentation with Comments	5.D	Assignment: *Documenting Classes with Javadoc
5.4 Accessor Methods	3.B, 5.B	
5.5 Mutator Methods	3.B, 4.B	
5.6 Writing Methods	1.B, 3.B	
5.7 Static Variables and Methods	3.B, 5.A	Lab 5: *Parity Functions
5.8 Scope and Access	3.B, 5.B	Lab 6: *Set Ops Lab 7: Three-Method Breakdown
5.9 `this` Keyword	2.C	
5.10 Ethical and Social Implications of Computing Systems		
Complete Personal Progress Checks for Unit 5		Personal Progress Check MCQ Part A Personal Progress Check MCQ Part B Personal Progress Check FRQ
Unit 5 Review		
Unit 5 Test		

Unit 6: Array		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
6.1 Array Creation and Access	1.C, 3.D	
6.2 Traversing Arrays	2.B, 3.D, 4.B	
6.3 Enhanced `for` Loop for Arrays	3.D, 4.C	
6.4 Developing Algorithms Using Arrays	1.B, 3.D, 5.D	Assignment: *Test Cases
Complete Personal Progress Checks for Unit 6		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 6 Review		
Unit 6 Test		

Unit 7: `ArrayList`		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
7.1 Introduction to `ArrayList`	1.B, 3.D	Assignment: Why Don't We Always Use This?
7.2 `ArrayList` Methods	2.C, 3.D	
7.3 Traversing ArrayLists	2.C, 3.D	
7.4 Developing Algorithms Using ArrayLists	3.D, 4.A	
7.5 Searching	3.D, 5.C	
7.6 Sorting	2.D	Assignment: *Practical Big-O Assignment: Insertion Sort Assignment: Selection Sort
7.7 Ethical Issues Around Data Collection		Assignment: *Computing Ethics
Complete Personal Progress Checks for Unit 7		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 7 Review		
Unit 7 Test		

Unit 8: 2D Array		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
8.1 2D Arrays	1.B, 1.C, 3.E	Lab 8: 2D Array Shifter
8.2 Traversing 2D Arrays	2.B, 2.D, 3.E, 4.A	Lab 9: Taxman Assignment: Star Trek Revisited
Complete Personal Progress Checks for Unit 8		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 8 Review		
Unit 8 Test		

Unit 9: Inheritance		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
9.1 Creating Superclasses and Subclasses	1.A, 3.B	Lab 10: Taxable and Non-taxable Lab 11: Foreign Investment
9.2 Writing Constructors for Subclasses	3.B, 5.A	
9.3 Overriding Methods	3.B, 5.D	
9.4 `super` Keyword	1.C, 3.B	
9.5 Creating References Using Inheritance Hierarchies	3.A, 5.B	
9.6 Polymorphism	3.A, 5.B	Assignment: `ArrayList` of Investments
9.7 `Object` Superclass	1.C, 3.B	Assignment: Auto-Test
Complete Personal Progress Checks for Unit 9		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 9 Review		
Unit 9 Test		

Unit 10: Recursion		
Topic	**Suggested Skills**	**Highlighted Assignments & Labs**
10.1 Recursion	1.B, 5.A	Lab 12: Almost the Largest Assignment: It Might As Well Be Recursion Assignment: Now It's Recursion
10.2 Recursive Searching and Sorting	2.C, 2.D	Lab 13: Recursion-a-palooza Assignment: Recursive Mergesort Assignment: Mergesort
Complete Personal Progress Checks for Unit 10		Personal Progress Check MCQ Personal Progress Check FRQ
Unit 10 Review		
Unit Test		

Select Assignment Descriptions

[CTP1] [CTP3] Lab 2—Rolling Dice: CR9 In this assignment, we want to write a program that inputs a specially formatted string that indicates how many times to roll dice with different numbers of sides, and uses random numbers to simulate a roll. For example, `1d8 + 2d6 + 3` means to simulate rolling an eight-sided die once, a six-sided die twice, then add all results together plus 3. **(Skills 3.A, 3.C)** CR6 Instead of attacking this dead-on though, we'll brainstorm ways to reduce this problem to simpler versions we can tackle first. **(Skill 1.A)** We'll make lists of each individual task in this assignment and see what additional Java library methods we'll need to investigate. **(Skill 1.C)** CR4

[CTP2] Practical Big-O: Students will be given a series a code examples and asked to determine the Big-O running time based on the input size. Examples will include single loops, nested loops, single and nested loops in series, and disguised nesting that occurs when code inside a loop calls other methods, including library methods. **(Skill 2.D)** CR5

[CTP2] Output or Trick: Throughout the year, students will be given code examples and asked to determine the output. In some cases, the answer is straightforward, e.g., a series of `if` statements with output that only requires understanding boolean expressions. In other cases, there's a trick. For example, a loop that looks like it adds ten Integer objects to an `ArrayList` but actually adds ten references to the same `Integer` object. **(Skills 2.B, 2.C)**

[CON] [CTP3] Lab 5—Parity Functions: CR9 Write a program with two static methods called `evenParity` and `oddParity`. These functions each take a `String` and return a `boolean`. The `evenParity` method should return `true` if the `String` represents a binary number with evenparity, and `false` otherwise. For example. `evenParity("1100101")` would return `true` but `evenParity("11001")` would return `false`. The `oddParity` method does the same, but of course for odd parity. **(Skills 3.A, 3.C)** CR3

For the sake of robustness, if the `String` contains anything other than `'0'` or `'1'` characters, it should return `false`. For example, `oddParity("2374")` would return false.

[VAR and MOD] [CTP3] Lab 6—Set Ops: CR9 Create your own `Set` class for storing and manipulating sets of `String` objects. The class should include methods to add a `String` to the set and to determine if a `String` is in the `Set`. The class should also include methods to find the intersection, union (with no duplicates), and difference of two sets. For example, `S1.intersect(S2);` should set `S1` to the intersection of `S1` and `S2`. **(Skills 3.A–3.C)** CR3 CR6

[CTP 4] Test Cases: After another programming assignment has been provisionally completed, students will be asked to design a series of test cases for testing another student's unseen implementation. **(Skill 4.A)** CR7

[IOC] Computing Ethics: CR3 Choose one question to argue for or against in a short essay, documenting all sources. We'll choose in class to make sure all questions are covered.

1. The Morality of Theft

Choose one of the following to answer in a brief essay (several paragraphs). Support your answer with sound reasoning.

A. Is it morally permissible to inflict a DDoS attack on a target you think is itself immoral?
B. Is digital theft "better" than physical theft?
C. Is it permissible to steal something you want if you could not otherwise afford it?

CR9
The syllabus must include titles and descriptions for at least two labs. Labs must be explicitly labeled.

If the course uses labs provided by College Board, titles must be included to satisfy this curricular requirement.

CR6
The syllabus must include a brief description of an assignment describing how students will engage with one skill in Computational Thinking Practice 3. Assignments must explicitly label which skill(s) they address.

CR4
The syllabus must include a brief description of an assignment describing how students will engage with one skill in Computational Thinking Practice 1. Assignments must explicitly label which skill(s) they address.

CR5
The syllabus must include a brief description of an assignment how students will engage with one skill in Computational Thinking Practice 2. Assignments must explicitly label which skill(s) they address.

CR3
The syllabus must include four student activities, each of which describes how it relates to one of the four big ideas. All of the big ideas must be represented. Each activity must be labeled with the related big idea(s).

2. The Duty of Developers

Choose one of the following to answer in a brief essay (several paragraphs). Support your answer with sound reasoning.

A. Do programmers have a responsibility to develop code that is as reliable as possible?
B. Do programmers have a responsibility to develop code that is as readable as possible?
C. Should programmers be held legally responsible for code failures in the way that doctors are held legally responsible for medical failures?

[CTP5] Documenting Classes with Javadoc: Each student is given a method to investigate. Students must explain the purpose of the method using Java documentation. **(Skill 5.A)** From there we will construct, as a class, our own method header documentation requirements for future programming assignments. **CR8**

CR7
The syllabus must include a brief description of an assignment describing how students will engage with one skill in Computational Thinking Practice 4. Assignments must explicitly label which skill(s) they address.

CR8
The syllabus must include a brief description of an assignment describing how students will engage with one skill (skill 5.A, 5.B, 5.C, or 5.D) in Computational Thinking Practice 5. Assignments must explicitly label which skill(s) they address.