

This syllabus was originally AP[®] Audit approved for AP[®] Computer Science AB.

This syllabus is provided courtesy of A+ Computer Science.

Updated 08/01/2010 to include Graphs, graph-theory, Radix Sort, and Heap Sort

www.apluscompsci.com

All labs, tests, quizzes, slides, and worksheets referenced on this syllabus are included in the A+ Computer Science Curriculum packages.

AP[®] Computer Science AB

Course Overview

I teach AP[®] Computer Science as a three year sequence and as a two year sequence. An introductory Computer Science course, which introduces many AP Computer Science A topics, is recommended, but not required as the first course in our sequence. Students can take AP Computer Science A after the introductory Computer Science course or take AP Computer Science A as their first Computer Science course. After completion of the AP Computer Science A course, students have the option to continue on and study Computer Science further by taking AP Computer Science AB. All students in the A course are encouraged to take the AB course regardless of the grade earned in the A course. I have found that many students have higher grades and greater success in the AB course as compared to the A course as the repetition and review of the A material in the AB course increases understanding and comprehension. Our curriculum for the AP Computer Science AB course includes all topics and course objectives for AP Computer Science A and AP Computer Science AB as described in the *AP Computer Science Course Description*. This course concentrates on Object Oriented Design, Data Structures, Software Development, and the building of a strong logic foundation, including heavy concentration on the AP Computer Science GridWorld Case Study. All students taking AP courses are encouraged to take the AP exam.

Lab Setup / Lab Usage

As with many other schools, we teach AP Computer Science in a computer lab. We have all PCs set against the outer wall of the room. The students work on lab assignments individually and collaboratively as the particular assignment dictates. Each student in each of my classes has access to a PC during class, before school, and after school. For each unit of coverage, I have multiple computer-based programming lab assignments. I have found that students do best when given many differing lab assignment options. The more options the students can choose from the higher the rate of student success. During new topic discussions, students open sample programs on the computer to see the topics in code form. Students are instructed to make changes to the sample programs and then run the programs to test the changes. Students have many opportunities to use computers. The word lab is used below in the syllabi to identify a computer programming assignment.

TIME	TOPICS – Covered in AP CS A
1 week	<p data-bbox="280 233 1365 268">Unit 0A – Computer Science, Computer Lab, and Objects Introduction</p> <p data-bbox="280 306 1490 485">AP Topics – Test classes and libraries in isolation; Identify and correct errors : compile-time, run-time, logic; Categorize error: compile-time, run-time, logic; Employ techniques such as using a debugger, adding extra output statements, or hand-tracing code; Understand and modify existing code; Inheritance; Object-oriented development; Top-down development; Encapsulation & information hiding.</p> <p data-bbox="280 522 1479 768">Student Objectives - Students will learn what Computer Science is, how a computer lab works, how to use the computer, how the network is setup, and how to use the labs and the network in an acceptable/ethical manner. Students will learn the basic syntax for Java and how to debug a program, the difference between a compile error and a syntax error, how to identify and correct errors, how to add to and remove from existing code. Students gain experience working with, modifying, and expanding a large program. Students will use a modern IDE (NetBeans / Eclipse) with a full slate of features, including a debugger.</p> <p data-bbox="280 795 1468 932"><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
1 week	<p data-bbox="280 938 594 974">Unit 0B - Computers</p> <p data-bbox="280 1012 1471 1152">AP Topics – Primary and secondary memory; processors; peripherals; language translators/compilers; virtual machines; operating systems; networks; single-user systems; networks; system reliability; privacy; legal issues and intellectual property; social and ethical ramifications of computer use.</p> <p data-bbox="280 1190 1463 1297">Student Objectives – Students will learn all of the fundamental components of a computer, how a computer works, hardware, software, compilers, programming languages, basic computer operations, integrity, and responsible use of the computer.</p> <p data-bbox="280 1335 1468 1436"><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Slides, Worksheets, etc.</i> <i>Assessments : Quizzes and Tests(m/c)</i></p>
1 week	<p data-bbox="280 1442 854 1478">Units A and B – Output and Variables</p> <p data-bbox="280 1516 1403 1623">AP Topics – Primitive types vs. Objects; Constant declarations; Variable declarations; Console output; Java library classes; Simple data types(int, boolean, double); Classes; Representations of numbers in different bases; Limitations of finite representations.</p> <p data-bbox="280 1661 1484 1768">Student Objectives – Students will learn what a variable is, how to define a variable, how to assign values to a variable, the difference between a primitive type and a reference, and how to print/println values to the console window.</p> <p data-bbox="280 1803 1468 1938"><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : ASCII Art, Area of a Triangle, Area of a Square</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>

<p>1 week</p>	<p>Units C and 1 – Input and Methods – Generic Classes</p> <p>AP Topics – Variable declarations; Console output; Java library classes; Simple data types(int, boolean, double); Classes; Method declarations, Class declarations; Parameter declarations.</p> <p>Student Objectives – Students will learn how to perform basic input operations, write methods, define and pass parameters, and use graphics to make shapes and pictures. Students will use Scanner, a Generic Java class, to perform input operations.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Smiley Face, House, Robot</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>2 weeks</p>	<p>Units 2 and 3 – Classes, OOP, Math Operations</p> <p>AP Topics – Object Oriented development; Top-down development; Encapsulation and information hiding; Procedural abstraction; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn how to declare a class, class methods, and parameters, the difference between constructors, accessors, and modifiers, learn how to read and understand a problem description, purpose, and goals. Students will learn to solve problems using mathematical operators(+, -, /, *, %), mathematical formulas, and Math class methods.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Area of Triangle, Area of Square, Heron’s Formula, Miles Per Hour, Quadratic Formula</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>1 week</p>	<p>Unit 4 - Strings and OOP</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn how to instantiate a String, more about references, how to create a reference to a String, perform String input and output, how to use String methods(length, substring, indexOf, charAt), how to write return methods(toString), and how to create more sophisticated classes.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Concatenation, FirstLastLetter, StringRipper, Name</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>

<p>3 weeks</p>	<p>Units 5-7 – Conditionals – If, If else, If else if, Switch Case</p> <p>AP Topics – Conditional; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn how to use if, if else, if else if, and switch case to test conditions and add decision making to their programs, and Boolean conditions and variables. Students learn how to use relational operators(>,<,>=,<=,!).</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : OddEven, Animated Car, Distance, SocialSecurity#, Grade, HexToBinary, Number Compare, String Equality, String Length, Morse Code, Mouse Button Tester</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>2 weeks</p>	<p>Units 8-9 – Iteration – For Loop and While Loop</p> <p>AP Topics – Iteration; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn how to use for loops, use while loops, add iterative processes to their programs, and use Boolean conditions and variables. Students learn the different parts of a loop and when to use a particular type of loop.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Greatest Common Divisor, Prime, MutlificationTable, Binary To Ten, Perfect, BackWards String, Box Word, Factorial, Decreasing Word, Divisors, Reverse Num</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>1 week</p>	<p>Unit 10 - Boolean Logic and Boolean Laws</p> <p>AP Topics – Boolean; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn boolean laws, truth tables, logical operators(&&, , !, ^), how to use do while loops, how to use boolean logic to solve problems, and how to use Random and Math.random() to generate random numbers.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Lab : Password Checker, BiggestDouble, BiggestString, Guessing Game</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>

<p>1 week</p>	<p>Unit 11 - Iteration – Nested Loops</p> <p>AP Topics – Iteration; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn how to use nested loops, add iterative processes to their programs, and use Boolean conditions and variables. Students learn how to use nested for and nested while loops.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : TriangleLetter, BoxWords, TriangleWords, RandomColoredBoxes, Triples</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>2 weeks</p>	<p>Units 12-13 – Chopping Strings and File Input</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn how to use Scanner to chop up Strings, to read data from data files, to instantiate Objects using the data extracted from files. Students learn more about constructor overloading and using a single class for multiple purposes.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Prime, Box, Biggest Number, Average, GCD, Dog Food, Line Breaker, Pig Latin</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>2 weeks</p>	<p>Units 14-15 - One dimensional arrays</p> <p>AP Topics – One-dimensional arrays; Traversals; Insertions; Deletions; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn how to instantiate a one-dimensional array, add items to a one-dimensional array, delete items from a one-dimensional array, and use a one-dimensional array to solve problems. Students will learn the differences between arrays of primitives and arrays of references.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Biggest, Histogram, GraphicTree, Fibonacci, WordSorter, StringArrays</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
	<p>End of Semester One</p>

<p>2 weeks</p>	<p>Unit 16 - Arrays</p> <p>AP Topics – One-dimensional arrays; Two-dimensional arrays; Traversals; Insertions; Deletions; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn how to instantiate a one-dimensional and two-dimensional array, add items to a one-dimensional and two-dimensional array, delete items from a one-dimensional and two-dimensional array, and use a one-dimensional and two-dimensional to solve problems.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs: Pascal’s Triangle, 3DArray, Images, Fancy Words([], X, etc.)</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 17 – References / Parameters</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn more about references and parameter passing. Students will learn the differences between passing primitives and references as parameters.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : ArrayTools, WordPrinter, LetterBoxes, TwoDRay, ThreeDRay</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>1 week</p>	<p>Unit 18 - Interfaces / OOP</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations.</p> <p>Student Objectives – Students will learn how to design and implement a class; apply data abstraction and encapsulation; and implement an interface and learn why interfaces are useful. Students will learn how interfaces are used to build hierarchies.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Monster, SortByVowels, RomanNumerals</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

<p>1 week</p>	<p>Unit 19 – Array of References</p> <p>AP Topics – One-dimensional arrays; Traversals; Insertions; Deletion; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn more about storing references in arrays. Students will learn the difference between arrays of primitives and arrays of references.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : ArrayOMonsters, AP CS GridWorld Case Study, GradeBook, TicTacToe</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free resposne)</i></p>
<p>2 weeks</p>	<p>Unit 20 - Inheritance</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations; Read and understand class specifications and relationships among the classes("is-a", "has-a" relationships); Understand and implement a class hierarchy; Identify reusable components from existing code using classes and class libraries; Choose appropriate data representation and algorithms.; Extend a class using inheritance.</p> <p>Student Objectives – Students will learn how to extend a given class using inheritance, design and implement a class hierarchy, write a multi-tiered game with graphics and animation. Students will learn how to build a new class from an existing class using extends and super calls.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, AP CS GridWorld Case Study, etc.</i> <i>Labs : Pong(Block, Ball, Paddle, Game), AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>2 weeks</p>	<p>Unit 21 – ArrayList / Generic Classes</p> <p>AP Topics – One-dimensional arrays; Traversals; Insertions; Deletions; Object Oriented development; Searching; Sorting; Test classes and libraries in isolation; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms.</p> <p>Student Objectives – Students will learn how to add to, delete from, sort, search, and perform all types of manipulations on an ArrayList.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, AP CS GridWorld Case Study, etc.</i> <i>Labs : GradeBook, Histogram, Map, AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

<p>2 weeks</p>	<p>Unit 22 - Abstract Classes</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations; Read and understand class specifications and relationships among the classes("is-a", "has-a" relationships); Understand and implement a class hierarchy; Identify reusable components from existing code using classes and class libraries; Choose appropriate data representation and algorithms; Extend a class using inheritance.</p> <p>Student Objectives – Students will learn how to design and implement an abstract class, extend an abstract class to make sub classes, and implement an interface. Students will learn to compare and contrast a class, an interface, and an abstract class. Students will learn when to use an interface and when to use an abstract class.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, AP CS GridWorld Case Study, etc.</i> <i>Labs : BlackJack(Card, Deck, Player, Dealer, Game), AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>2 weeks</p>	<p>Unit 23 - GridWorld Case Study</p> <p>AP Topics – Object Oriented development; Class declarations; Interface declarations; Read and understand class specifications and relationships among the classes("is-a", "has-a" relationships); Identify reusable components from existing code using classes and class libraries; Choose appropriate data representation and algorithms.; Extend a class using inheritance; Pre and post conditions; Assertions.</p> <p>Student Objectives – Students will learn how to modify and extend large pre-written programs. Students will learn more about the case study and perform modifications to many of the case study classes as well as creating new classes. Students will learn more interfaces, abstract classes, and inheritance.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, AP CS GridWorld Case Study, etc.</i> <i>Labs : AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 24 - Recursion</p> <p>AP Topics – Recursion; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Class declarations.</p> <p>Student Objectives – Students will learn how to use recursion to solve problems, the benefits of using recursion, when to use recursion, and the negative effects of using recursion.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : RecursiveCircles, MazeSolver, TrashCollector</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

<p>1 week</p>	<p>Unit 25 - Advanced Sorting and Searching / Comparable</p> <p>AP Topics – One-dimensional arrays; Traversals; Insertions; Deletions; Object Oriented development; Searching; Sequential Search; Binary Search; Sorting; Selection Sort; Insertion Sort; Merge Sort; Test classes and libraries in isolation; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts.</p> <p>Student Objectives – Students will learn to identify all sorting and searching algorithms, code all sorting and searching algorithms, and to select the appropriate sorting and searching algorithm for the appropriate situation. Students will learn where to use a particular sort/search and the benefits of using a particular type of sort/search.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Insertion Sort, QuickSort, MergeSort</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>3 weeks</p>	<p>AP Review Time</p> <p><i>Guided Practice : Review topics, AP CS GridWorld Case Study</i> <i>Guided Practice : Past year’s free response and multiple choice questions</i> <i>Guided Practice : Slides, AP CS GridWorld Case Study</i> <i>Readings : Past year’s free response and multiple choice questions</i> <i>Readings : Review book units</i></p>
	<p>End of Semester Two</p> <p>All topics listed above are covered in AP CS A.</p> <p>All topics covered in CS AP A are reviewed in AP CS AB.</p>

TIME	New Topics Not Covered in AP A
1 week and ongoing during the entire AP CS AB year	<p>Unit 1A – AP Computer Science A Review(Topics Listed on AP CS A syllabi)</p> <p>AP Topics – Test classes and libraries in isolation; Identify and correct errors : compile-time, run-time, logic; Categorize error: compile-time, run-time, logic; Employ techniques such as using a debugger, adding extra output statements, or hand-tracing code; Primary and secondary memory; processors; peripherals; language translators/compiler; virtual machines; operating systems; networks; single-user systems; networks; system reliability; privacy; legal issues and intellectual property; social and ethical ramifications of computer use.</p> <p>Student Objectives - Students will review what Computer Science is, how a computer lab works, how to use the computer, how a computer works, how the network is setup, and how to use the labs and the network in an acceptable/ethical manner. Students will review AP CS A topics. Students will review Object Oriented programming and Java fundamentals.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Magic Squares, AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
1 week	<p>Unit 1B – Matrices</p> <p>AP Topics – One-dimensional arrays; Two-dimensional arrays; Traversals; Insertions; Deletions; Object Oriented development; Class design; Method declarations; Parameter declarations; Class declarations; Top-down development; Encapsulation & information hiding.</p> <p>Student Objectives – Students will learn how to instantiate a one-dimensional and two-dimensional array, add items to a one-dimensional and two-dimensional array, and delete items from a one-dimensional and two-dimensional array.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs: Pascal’s Triangle, 3DArray, Tic Tac Toe</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
2 weeks	<p>Unit 2 - Interfaces / Inheritance / Abstract Classes</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Specify the purpose and goals for a problem; Decompose a problem into classes; define relationships and responsibilities of those classes; Design and implement a set of interacting classes; design an interface; choose appropriate advanced data structures and algorithms; Understand and modify existing code; Inheritance; Object-oriented development.</p> <p>Student Objectives – Students will learn how to extend a given class using inheritance, design and implement a class hierarchy, and write a multi-tiered game with graphics and animation. Students will use interfaces, abstract classes, and inheritance to write a multi-class project.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Labs : Pong, AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

<p>1 week</p>	<p>Unit 3 – References / Parameters</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn more about references and parameter passing. Students will learn the differences between passing primitives and references as parameters. Students will use loops to search lists and Strings for specified values. Students will learn how to use regular expressions to remove and locate values within a String.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : WordPrinter, TwoDRay, ThreeDRay</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>1 week</p>	<p>Unit 4 - ArrayList / Generic Classes</p> <p>AP Topics – One-dimensional arrays; Traversals; Insertions; Deletions; Object Oriented development; Searching; Sorting; Test classes and libraries in isolation; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Big-Oh notation.</p> <p>Student Objectives – Students will learn how to add to, delete from, sort, search, and perform all types of manipulations on an ArrayList. Students will learn more about Generic classes.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : GradeBook, Histogram, Map, AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Units 5 - Iterators</p> <p>AP Topics – Iterators; ListIterators; Interfaces; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn about Iterators and ListIterators. Students learn when to use iterators and why to use iterators. Students learn the differences between an Iterator and a ListIterator.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : RemoveIt, ReplaceIt, CountsOdds, CountEvens</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>

<p>1 week</p>	<p>Unit 6 - Interfaces / Comparable</p> <p>AP Topics – Interfaces; Designing a hierarchy; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations.</p> <p>Student Objectives – Students will learn how will design and implement a class; apply data abstraction and encapsulation; and implement an interface and why interfaces are useful. Students will learn how the Comparable interface is used to create a hierarchy. Students will also combine interfaces, abstract classes, and inheritance to write a multi-class project.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Sorting By Vowels, Auto Parts, TV Shows, Tic Tac Toe, GradeBook</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 7 – Sets / Generic Classes</p> <p>AP Topics – Set; TreeSet; HashSet; Iterators; Traversals; Insertions; Deletions; Object Oriented development; Identify boundary cases and generate appropriate test data; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts; throw runtime exceptions; Big-Oh notation.</p> <p>Student Objectives – Students will learn more about the Java Collection Framework, how to use a HashSet, how to use a TreeSet, and the differences between HashSet and TreeSet. Runtime specifications will be discussed.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Lab : Odds, Evens, Union/Intersection/Difference</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>1 week</p>	<p>Unit 8 – Maps / Generic Classes</p> <p>AP Topics – Map; TreeMap; HashMap; Iterators; Traversals; Insertions; Deletions; Object Oriented development; Identify boundary cases and generate appropriate test data; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts; throw runtime exceptions; Big-Oh notation.</p> <p>Student Objectives – Students learn about the Map interface and the Map hierarchy. Students will learn about the TreeMap class, the HashMap class, and runtime specifics for each.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Histogram, Relatives, Translation, AutoParts, MorseCode, EZProgramming</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>

<p>2 weeks</p>	<p>Units 9-10 - Recursion</p> <p>AP Topics – Recursion; Maps; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Class declarations.</p> <p>Student Objectives – Students will learn how to use recursion to solve problems, the benefits of using recursion, when to use recursion, and the negative effects of using recursion. Students will design recursive solutions to simple math functions and to complex connection problems. Students will learn the runtime efficiency of all algorithms created. Students will combine Maps and Recursion to solve graph-theory problems. Students will learn the different terms associated with graphs and graph-theory. Students will use depth-first searches, brute-force, greedy, and divide-and-conquer algorithms. Students will analyze the A* and Dijkstra algorithms to understand shortest path solutions and heuristics. Students will learn how to create greedy algorithms to create combinations, permutations, and perform pattern-matching.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : GCF, CellCounter, MazeSolver, Garbage Collector, WordFun, Connections</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Units 11 – Number Systems / Base Conversion</p> <p>AP Topics – Converting Numbers To Different Bases; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students will learn how to convert numbers between different bases, more about binary numbers, and how computers use electricity to do what they do.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : BaseConversion, WordToBinary, BinarySorter</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 12 - Boolean Logic and Boolean Laws</p> <p>AP Topics – Boolean; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.</p> <p>Student Objectives – Students learn boolean laws, truth tables, how to use do while loops, how to use boolean logic to solve problems, and how to use Random and Math.random().</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Lab : PasswordChecker, GuessingGame, Triples, TicTacToe</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>

<p>1 week</p>	<p>Unit 13 - Stacks / Generic Classes</p> <p>AP Topics – Stacks; One-dimensional arrays; Lists; Traversals; Insertions; Deletions; Object Oriented development; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts.</p> <p>Student Objectives – Students will learn to design and implement a Stack, use a predefined Stack, and use Stacks to solve problems.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : ExpressionSolver, SyntaxChecker, MakeAStack</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 14 - Queues / Generic Classes</p> <p>AP Topics – Queues; One-dimensional arrays; Lists; Interfaces; Traversals; Insertions; Deletions; Object Oriented development; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts.</p> <p>Student Objectives – Students will learn to design and implement a Queue, use a predefined Queue, and use Queues to solve problems.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : PalinList, PriorityQueue, MonsterQueue, MakeAQueue</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
	<p>End of Semester One</p>

<p>2 weeks</p>	<p>Unit 15 – Linked Lists</p> <p>AP Topics – LinkedLists(singly, doubly, circular); Traversals; Insertions; Deletions; Iterators; Object Oriented development; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts; Big-Oh notation.</p> <p>Student Objectives – Students will learn how to design and implement a LinkedList using ListNode, use the Java LinkedList class, and how to store Objects in a ListNode inside of a LinkedList. Runtime efficiency is discussed and compared to that of an array.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : LinkedListFunHouse1, LinkedListFunHouse2, HistogramListOne, HistogramListTwo</i> <i>Assessments : Labs, Quizzes, and Tests(m/c)</i></p>
<p>1 week</p>	<p>Unit 16 – Hash Tables</p> <p>AP Topics – HashTables; Hashing; Traversals; Insertions; Deletions; Object Oriented development; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts.</p> <p>Student Objectives – Students will learn how will design and implement a HashTable using arrays and linked lists. Students will learn the importance of writing correct hashCode and equals methods for all Objects being stored in the HashTable. Students will learn about linear-probing, bucket-hashing, chaining, and collisions.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Hash Table of Numbers, Hash Table of Words</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

<p>2 weeks</p>	<p>Unit 17 – Trees / Generic Classes</p> <p>AP Topics – Trees; Binary Trees; Traversals; Insertions; Deletions; Object Oriented development; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts; Big-Oh notation.</p> <p>Student Objectives – Students will learn how to design and implement a BinarySearchTree using TreeNode and how to store Objects in a TreeNode inside of a BinarySearchTree. Runtime efficiency is discussed and compared to that of a LinkedList. Students will use depth-first traversals using recursion and breadth-first traversals using stacks and queues. Students will implement a Generic tree class. Students will learn about Huffman coding and create a Huffman tree.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Basic Tree, Histogram Tree, Various Tree Fun</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free resposne)</i></p>
<p>1 week</p>	<p>Unit 18 – Heaps / Priority Queues</p> <p>AP Topics – Heaps; Trees; Priority Queues; Arrays; Traversals; Insertions; Deletions; Object Oriented development; Searching; Identify boundary cases and generate appropriate test data; Choose appropriate data representation and algorithms.</p> <p>Student Objectives – Students will learn how to add to, delete from, search, and perform all types of manipulations on a Heap and PriorityQueue. Students will combine PriorityQueues and the A* algorithm to design algorithms.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Heap, MonsterQueue, XQueue</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

<p>1 week</p>	<p>Unit 19 - Advanced Sorting and Searching / Comparable</p> <p>AP Topics –Traversals; Insertions; Deletions; Object Oriented development; Searching; Sequential Search; Binary Search; Sorting; Selection Sort; Insertion Sort; Merge Sort; Identify boundary cases and generate appropriate test data; Perform integration testing; Choose appropriate data representation and algorithms; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts; Big-Oh notation.</p> <p>Student Objectives – Students will learn to identify all sorting and searching algorithms, code all sorting and searching algorithms, and to select the appropriate sorting and searching algorithm for the appropriate situation. Students will learn how to identify the divide-and-conquer algorithms.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Insertion Sort, QuickSort, MergeSort, SortTimer</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 20 – Big O Notation</p> <p>AP Topics – Big-Oh notation; Worst-case and average-case time and space analysis; Sets; Maps; Linked Lists; Arrays; Trees; Sequential Search; Binary Search; Sorting; Selection Sort; Insertion Sort; Merge Sort; Identify boundary cases and generate appropriate test data; Analysis of algorithms; Informal comparisons of running times; Exact calculation of statement execution counts.</p> <p>Student Objectives – Students will learn to identify all sorting and searching algorithms, code all sorting and searching algorithms, and to select the appropriate sorting and searching algorithm for the appropriate situation. Students will learn appropriate BigO notation for all Java Collections, Trees, Lists, Heaps, Hash Tables, Graphs, Sorts, and Searches.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : Various</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>

2 weeks

Unit 21 - GridWorld Case Study

AP Topics – Object Oriented development; Class declarations; Interface declarations; Read and understand class specifications and relationships among the classes("is-a", "has-a" relationships); Identify reusable components from existing code using classes and class libraries; Choose appropriate data representation and algorithms; Extend a class using inheritance; Pre and post conditions; Assertions; Understand and implement a class hierarchy; Specify the purpose and goals for a problem; Decompose a problem into classes; define relationships and responsibilities of those classes; Design and implement a set of interacting classes; design an interface; choose appropriate advanced data structures and algorithms.

Student Objectives – Students will learn how to modify and extend large pre-written programs. Students will learn how to add appropriate data structures to the case study. Students will learn how to apply software engineering techniques while designing and implementing a GridWorld game. Students will learn about testing fundamentals and validation. Students will create a test plan that includes objectives and test cases for the finished product.

Guided Practice : Topic discussion, Example program analysis and modification, Worksheets

Readings : Labs, Slides, Worksheets, etc.

Labs : AP CS GridWorld Case Study Labs

Assessments : Labs, Quizzes, and Tests(m/c and free response)

<p>2 weeks</p>	<p>Unit 22 - Abstract Classes / Inheritance</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations; Read and understand class specifications and relationships among the classes("is-a", "has-a" relationships); Understand and implement a class hierarchy; Specify the purpose and goals for a problem; Decompose a problem into classes; define relationships and responsibilities of those classes; Design and implement a set of interacting classes; design an interface; choose appropriate advanced data structures and algorithms; Top-down development; Encapsulation & information hiding.</p> <p>Student Objectives – Students will learn how to design and implement an abstract class, extend an abstract class to make sub classes, and implement an interface. Students will learn how to apply software engineering techniques while designing and implementing a game. Students will learn about testing fundamentals and validation. Students will create a test plan that includes objectives and test cases for the finished product.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : BlackJack, AP CS GridWorld Case Study</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>1 week</p>	<p>Unit 23 – Interfaces / Inheritance / Abstract Classes</p> <p>AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations; Read and understand class specifications and relationships among the classes("is-a", "has-a" relationships); Understand and implement a class hierarchy; Specify the purpose and goals for a problem; Decompose a problem into classes; define relationships and responsibilities of those classes; Design and implement a set of interacting classes; design an interface; choose appropriate advanced data structures and algorithms; Top-down development; Encapsulation & information hiding.</p> <p>Student Objectives – Students will learn how to design and implement an abstract class, extend an abstract class to make sub classes, and implement an interface.</p> <p><i>Guided Practice : Topic discussion, Example program analysis and modification, Worksheets</i> <i>Readings : Labs, Slides, Worksheets, etc.</i> <i>Labs : StarFighter</i> <i>Assessments : Labs, Quizzes, and Tests(m/c and free response)</i></p>
<p>3 weeks</p>	<p>Final Project</p> <p><i>Guided Practice : Review data structures, design processes, and tools available</i> <i>Assessments : Design and build a project that encompasses components from the entire year. Follow the provided rubric.</i></p>
	<p>End of Semester Two</p>

Teaching Strategies

Topics are broken down into manageable pieces. Each topic is introduced and discussed in a group setting. Sections of each topic are discussed and then reinforced using example computer programs. Students run the example programs, make changes to the programs, and ask questions about the programs. Students are presented with many examples and explanations for all topics presented. Worksheets are provided that enhance the discussions and provide students the opportunity to practice the concepts without having to use the computer. Students are encouraged to use the computer to test their answers on the worksheets. Lab time is provided so that each student has the opportunity to apply the concepts in a hands-on situation using a PC. For each topic, there are many computer-based programming lab assignments so that each student has the opportunity to practice the topic in different ways. Quizzes are given to provide feedback and to gain information about the learning process. Tests are given in multiple-choice and free response format in a way that models what students will see on the AP test.

Teacher Resources

Armstrong, Stacey. *A+ Computer Science: Computer Science Curriculum Solutions*. <http://apluscompsci.com>, 2006

The College Board. *AP GridWorld Case Study*. New York: College Entrance Examination Board, 2006.

Teukolsky, Roselyn. *Barron's AP Computer Science Levels A and AB 2007*, 3rd ed. Hauppauge, N.Y.: Barron's Educational Series, 2006.

LIST ANY MORE RESOURCES/BOOKS THAT YOU USE

This syllabus was originally AP[®] Audit approved for AP[®] Computer Science AB.

This syllabus is provided courtesy of A+ Computer Science.

Updated 08/01/2010 to include Graphs, graph-theory, Radix Sort, and Heap Sort

www.apluscompsci.com

All labs, tests, quizzes, slides, and worksheets referenced on this syllabus are included in the A+ Computer Science Curriculum packages.