

AP[®] Computer Science AB

Syllabus 1

Course Description

In accordance with the College Board's course description, AP[®] Computer Science AB is a second course in computer science taught in Java. AP Computer Science A (or equivalent) is a prerequisite. The course incorporates all of the topics listed for AP Computer Science AB in the Topic Outline, including the GridWorld case study and review of AP Computer Science A topics.

Development of useful computer programs and classes is used as a context for the formal, in-depth study of computer science concepts emphasizing abstraction, design, development and use of data structures, and study of standard algorithms and typical applications. Throughout the course, object-oriented methodology and design are emphasized to make programs understandable, adaptable, and, when appropriate, reusable. A large part of the course is built around problem solving and developing computer programs or parts of programs that correctly solve a given problem.

All topics from AP Computer Science A are reviewed and expanded. The GridWorld case study is used for review of A-level topics and expanded to include applications of newly taught data structures and algorithms. In addition, an understanding of the basic hardware and software components of computer systems and their responsible use are integral parts of the course. Not only are ethical and social implications emphasized at the beginning of the course, but they also are addressed whenever opportunities arise throughout the year. For example, relevant topics/articles are discussed as they are reported in the media. Privacy, intellectual property, and responsible use of computers are recurring themes throughout the year.

Instructional strategies range from traditional lectures to the development of useful computer programs and also include guided programming activities, role-play, simulations, diagramming, class discussions, written descriptions, comparing and contrasting data structures, and algorithms. Both independent and cooperative work is part of the course.

Assessment of student learning includes detailed evaluation of programming projects and exercises, as well as tests, paper-and-pencil exercises, written comparisons and summaries, and class activities. Programming projects include both guided explorations and more open-ended programming assignments to solve a given problem. Much of the students' work is assessed using rubrics, including rubrics similar to those used in reading AP Exams.

Resources

Computers

Each student has individual access to a computer at all class times, during study halls, and after school. Each computer is running the current version of Java with a current IDE.

Texts (all students have a copy)

Java Software Structures for AP Computer Science AB, John Lewis, Joseph Chase, and Leigh Ann Sudol, 2005, Addison-Wesley/Pearson Education, Inc., Boston (referenced as LCS in units below).

Java Methods: An Introduction to Object-Oriented Programming, Maria Litvin and Gary Litvin, 2001, Skylight Publishing, Andover, MA (referenced as Litvin-A in units below—for APCS-A topics).

AP GridWorld Case Study, 2006, The College Board, New York.

Java Methods AB: Data Structures, Maria Litvin and Gary Litvin, 2003, Skylight Publishing, Andover, MA (referenced as Litvin-AB in unit below).

Review Materials

(all students have a copy when they are being used)

- Released questions from previous AP Computer Science AB Exams
- *Addison-Wesley's Review for the AP Computer Science Exam in Java*, 2nd Edition, Susan Horwitz and Leigh Ann Sudol, 2007, Pearson Educational, Boston.
- *Be Prepared for the AP Computer Science Exam in Java*, Maria Litvin, 2003, Skylight Publishing, Andover, MA.
- *5 Steps to a 5 AP Computer Science*, Kathleen Larson and David Levine, 2005, McGraw-Hill, New York.

Supplementary resources include but are not limited to:

- Supplements for Java 5 by materials from: *Java Methods A & AB: Object-Oriented Programming and Data Structures*, Maria Litvin and Gary Litvin, 2006, Skylight Publishing, Andover, MA.
- Java documentation at <http://java.sun.com>.
- *Sorting Out Sorting* video, 1981, Information Commons, University of Toronto, Toronto.
- Sorting and data structure visualizations such as those at: *Interactive Data Structure Visualizations*, Duane J. Jarc, University of Maryland, <http://nova.umuc.edu/~jarc/idsv>.

Major Units of Study

Major units are listed below along with the approximate timing, generally in the sequence in which they are presented to students. The Curricular Requirements for the AP Course Audit are referenced in the last column along with the College Board’s AP Computer Science Topic Outline.

Texts abbreviated as: LCS (Lewis, Chase, and Sudol)
 Litvin-A (Java Methods)
 Litvin-AB (Java Methods AB)

Major Units [Time]	Topics [C2]	Resources, Labs & Assessments	C2—The course includes all of the topics listed in the “Computer Science AB” column of the Topic Outline in the <i>AP Computer Science Course Description</i> .
1. Review and expand upon APCS-A Topics [6 weeks]	<ul style="list-style-type: none"> • Basic Computer Processing, Hardware, Software, Network, & Internet Components <ol style="list-style-type: none"> 1. Processors & operating systems 2. Language translator vs. compiler 3. JVM <ul style="list-style-type: none"> - class files & bytecode - cross-platform - applets vs. applications [C8] • Appropriate Use & Privacy <ol style="list-style-type: none"> 1. School & class expectations 2. Social & ethical <ul style="list-style-type: none"> - piracy, intellectual property - economic impact [C9] • Computer Number Systems <ol style="list-style-type: none"> 1. Conversions and arithmetic • Java programming basics <ol style="list-style-type: none"> 1. Data types: primitives & objects 2. Syntax & style 3. Booleans, conditionals, loops <ul style="list-style-type: none"> - time analysis of operations - loop invariant 4. Anatomy of classes & methods <ul style="list-style-type: none"> - decomposition into classes 5. Math, string, scanner classes 6. Random numbers 	<p>Resources: Litvin-A Ch. 1-12 (as review material) Java 5 handouts</p> <p>Labs and Assessments (to review, clarify and amplify previously taught material) (timing & assignments may be adjusted depending on student weaknesses during review):</p> <ul style="list-style-type: none"> • Selected Ramblecs Labs • Pie Chart Lab with calculations • Snack Bar (Vending Machine Class) Lab using multiple classes & objects • Equation Solver Lab—parsing strings & convert to numbers • Pig Latin Program—string manipulation • Rotate Array Lab—array manipulation • Plot results of sorts speeds from the Benchmark Lab • ArrayList<E> MergeSort Lab—implement the mergesort algorithm using an ArrayList [C3] 	<p>C8—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.</p>
			<p>C9—The course teaches students to recognize the ethical and social implications of computer use.</p>
			<p>C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.</p>

Major Units [Time]	Topics	Resources, Labs & Assessments	
	<ul style="list-style-type: none"> 7. Interfaces <ul style="list-style-type: none"> - comparable [C6] • Arrays, ArrayLists with Generics <ol style="list-style-type: none"> 1. Traversals, insertions, deletions 2. Space & time analysis, Big-Oh comparison of operations • 2-Dimensional Arrays <ol style="list-style-type: none"> 1. Traversals, insertions, deletions 2. Space & time analysis, Big-Oh comparison of operations • Searches—Binary & Linear <ol style="list-style-type: none"> 1. Space & time analysis 2. Big-Oh comparisons 3. Best/average cases • Sorts—Insertion, Selection, Merge & Quicksort algorithms <ol style="list-style-type: none"> 1. Space & time analysis 2. Big-Oh comparisons 3. Best/average cases • Common Errors <ol style="list-style-type: none"> 1. Compile-time, run-time, logic 2. Bounds testing 3. Debugging techniques 	<ul style="list-style-type: none"> • Written summaries and comparisons of searches and sorts • Tests: Chapters 1–6, Chapters 7–9, Chapters 10–12 	<p>C6—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 Appendices A and B of the <i>AP Computer Science Course Description</i>. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)</p>
<p>2. Object-Oriented Concepts in Java [2 weeks]</p>	<ul style="list-style-type: none"> • Software development & ethics <ol style="list-style-type: none"> 1. Development cycle 2. Correctness • UML Unified Modeling Language • Analysis of Algorithms <ol style="list-style-type: none"> 1. Growth functions & Big-Oh 2. Analyzing Loop execution 	<p>Resources: LCS Ch. 1–2</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • SmallInteger Class Lab—create and test an immutable data type • Math class and Random Number Lab • Exceptions Lab using previously defined SmallInteger class [C3] 	<p>C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.</p>

Major Units [Time]	Topics	Resources, Labs & Assessments
	<ul style="list-style-type: none"> • Classes <ol style="list-style-type: none"> 1. Constructors & methods <ul style="list-style-type: none"> - overloading - static modifier • Encapsulation <ol style="list-style-type: none"> 1. Visibility modifiers • Objects <ol style="list-style-type: none"> 1. State & behavior 2. References & aliases • Inheritance <ol style="list-style-type: none"> 1. Derived classes 2. Super reference 3. Overriding methods 4. Shadowing variables • Class Hierarchies <ol style="list-style-type: none"> 1. Abstract classes 2. Interfaces • Polymorphism • Exceptions <ol style="list-style-type: none"> 1. Throw exception 2. Try & catch statements 3. Exception Class Hierarchy 	<ul style="list-style-type: none"> • Test: Chapters 1 & 2
3. Java Collections, Sets, Iterators [2 weeks]	<ul style="list-style-type: none"> • Java Collections Framework • Set Collection <ol style="list-style-type: none"> 1. Interfaces 2. Iterators 3. Exceptions 4. Implemented with arrays <ul style="list-style-type: none"> - analysis of time & space [C6] 	Resources: LCS Ch. 3 Labs and Assessments: <ul style="list-style-type: none"> • Implement a set using an <code>ArraySet<E></code> of <code>BingoBalls</code> objects and align both interface and class codes to AP subset of <code>Set<E></code> and <code>Iterator<E></code> interfaces [C3] • Test: Collections, Sets, Iterators and Implementations

C6—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 Appendices A and B of the *AP Computer Science Course Description*. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)

C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

Major Units [Time]	Topics	Resources, Labs & Assessments
4. Linked Structures [3 weeks]	<ul style="list-style-type: none"> • Linked List <ol style="list-style-type: none"> 1. ListNode 2. Head & tail 3. External references 4. Insert 5. Delete 6. Traverse & iterate 7. Singly & doubly linked list 8. Circularly linked list 9. Common errors 10. Space analysis (vs. array) 11. Time analysis of operations <ul style="list-style-type: none"> - Big-Oh comparisons - best & average cases • Data structures using linked lists <ol style="list-style-type: none"> 1. Set implementation [C4] 	<p>Resources: LCS Chapter 4</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Basic Linked List Program using ListNode class (in small groups) —insert, delete & traverse implemented • AssassinManager Program using ListNode class with some use of Set<E> to play the interactive Assassin game • Stormy Seas Crime Scene Investigation Program—design & implement a program to find the final data pattern using a circularly linked list (ListNode) [C3] [C4] • Written comparisons of linear and binary search techniques and efficiency considerations using arrays and linked lists [C5] • Test: Linked lists implemented using ListNode
5. Lists and java.util. linkedList[C6] [1.5 weeks]	<ul style="list-style-type: none"> • List ADT <ol style="list-style-type: none"> 1. Unordered lists 2. Ordered lists 3. Indexed lists • List Interface & Linked List Class <ol style="list-style-type: none"> 1. java.util.List<E> interface 	<p>Resources: LCS Ch. 10</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Movie Actor Index Program using java.util.LinkedList<E> searching & traversing two LinkedLists via iterators

C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

C4—The course teaches students to use and implement commonly used algorithms and data structures.

C5—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

Major Units [Time]	Topics	Resources, Labs & Assessments	
	<ul style="list-style-type: none"> 2. java.util.LinkedList<E> <ul style="list-style-type: none"> - compared to ListNode class - head & tail - external references - insert - delete - traverse & iterate 3. Common errors 4. Space analysis [C4] 5. Time analysis of operations <ul style="list-style-type: none"> - Big-Oh comparisons - best & average cases - compared to ArrayList [C5] • Traversals <ul style="list-style-type: none"> 1. Iterator methods 2. ListIterator methods 	<ul style="list-style-type: none"> • Test: Lists, efficiencies and java.util. LinkedList<E> [C3] [C6] 	<p>C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.</p>
<p>6. Maps [1.5 weeks]</p>	<ul style="list-style-type: none"> • Map Interface java.util. Map<K,V> <ul style="list-style-type: none"> 1. Key and value pairs 2. Key set (java.util.Set<E>) 3. Implementation considerations 4. Insert 5. Delete 6. Traverse <ul style="list-style-type: none"> - iterate key set 7. Time and space analysis • Hash table <ul style="list-style-type: none"> 1. Hash code & hash functions <ul style="list-style-type: none"> - equals 2. Collisions <ul style="list-style-type: none"> - chaining - linear probing 3. Searching <ul style="list-style-type: none"> - time analysis 4. java.util.HashSet<E> 5. java.util.HashMap<K,V> [C4] [C6] 	<p>Resources: LCS Ch. 5</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • HashSet and HashMap Lab to investigate the differences in the data storage, insertion, deletion, and traversal of the two data types [C5] • Hash tables and Hash searching worksheets • Test: Maps and hashing 	<p>C4—The course teaches students to use and implement commonly used algorithms and data structures.</p>
			<p>C5—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.</p>
			<p>C6—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 Appendices A and B of the <i>AP Computer Science Course Description</i>. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)</p>

Major Units [Time]	Topics	Resources, Labs & Assessments
	<ul style="list-style-type: none"> • Time Analysis of Operations <ol style="list-style-type: none"> 1. Big-Oh comparisons 2. Best & average cases 3. Compared to other ADTs [C5] 	
7. Recursion, Searching & Sorting [2 weeks]	<ul style="list-style-type: none"> • Recursion (review) <ol style="list-style-type: none"> 1. In math 2. Infinite 3. Compared to iteration 4. Direct and indirect 5. Tower of Hanoi 6. Maze as an application 7. Efficiency 8. Hardware stack • Linear & Binary Searches <ol style="list-style-type: none"> 1. Algorithms 2. Efficiencies 3. Compared to hashing search • Sorts <ol style="list-style-type: none"> 1. Selection, Insertion, Merge & Quicksort 2. Algorithms 3. Comparing Sorts 4. O() Efficiency, best, average & worst cases [C4] [C5] [C6] 	<p>Resources: LCS Ch. 6–7</p> <ul style="list-style-type: none"> • <i>Sorting Out</i> Sorting video • Jarc’s Interactive Data Structure Visualization at http://nova.umuc.edu/~jarc/idsv <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • E-mail Address Book Lab—using java.util.HashSet<E>, java.util.HashMap<K,V> & recursion to expand an alias, plus choose & implement an appropriate sorting algorithm to alphabetically sort the resulting ArrayList of e-mail addresses • Test: Recursion, searching & sorting
8. Stacks & Queues [2 weeks]	<ul style="list-style-type: none"> • Stacks <ol style="list-style-type: none"> 1. Implementations of Stacks <ul style="list-style-type: none"> - java.util.Stack<E> 2. Insert 3. Delete 4. Time and space analysis • Hardware Stack <ol style="list-style-type: none"> 1. Recursion & its simulation • Queues <ol style="list-style-type: none"> 1. Implementations of Queues <ul style="list-style-type: none"> - java.util.Queue<E> 2. Insert 	<p>Resources: LCS Ch. 8–9 (except sect. 9.8)</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • PostFix Evaluation Lab using java.util.Stack<E> • Munch Game Lab—Queue implemented using linked list • Written data structures comparisons/summaries, including time & space analysis • Test: Stacks & queues

C5—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

C4—The course teaches students to use and implement commonly used algorithms and data structures.

C6—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 Appendices A and B of the *AP Computer Science Course Description*. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)

Major Units [Time]	Topics	Resources, Labs & Assessments
	<ul style="list-style-type: none"> 3. Delete 4. Time and space analysis 5. Radix sort • Postfix Notation and Evaluation • Maze as Application of Both <ul style="list-style-type: none"> 1. Depth vs. breadth first • Simulations using stack & queue • Compare/contrast stacks & queues [C4] [C5] 	
9. Trees [4 weeks]	<ul style="list-style-type: none"> • Trees <ul style="list-style-type: none"> 1. Terminology & tree types 2. Diagrams 3. Using recursion 4. Insert 5. Delete 6. Traversals <ul style="list-style-type: none"> - inorder, preorder, postorder 7. Time and space analysis • Implementations <ul style="list-style-type: none"> 1. TreeNode class implementation 2. java.util.TreeSet<E> 3. java.util.TreeMap<K,V> 4. Iterators • Tree Uses <ul style="list-style-type: none"> 1. Expression trees <ul style="list-style-type: none"> - inorder, preorder, postorder - infix, prefix, postfix evaluation 2. Binary search tree <ul style="list-style-type: none"> - ordering using Comparable [C4] [C6] 	<p>Resources: LCS Ch. 11–12 Tree animations/visualizations</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Binary Search Tree program using TreeNode—Design, implement & test a complete program to insert, delete and traverse a BST • Movies Program using TreeSet<E> to insert, delete and traverse alphabetically • MorseCode Lab using TreeNode to store code patterns • Diagramming tree traversals and expression trees [C3] [C4] • Evaluating Infix, Prefix and Postfix Expressions Worksheet • Test: Trees with TreeNode, TreeSets and TreeMaps

C4—The course teaches students to use and implement commonly used algorithms and data structures.

C5—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

C6—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 Appendices A and B of the *AP Computer Science Course Description*. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)

Major Units [Time]	Topics	Resources, Labs & Assessments
10. Heaps & Priority Queues [2 weeks]	<ul style="list-style-type: none"> • Heaps <ol style="list-style-type: none"> 1. Diagram as a tree 2. Array implementation 3. MinHeap <ul style="list-style-type: none"> - heap & reheap algorithms • Heapsort • Priority queues <ol style="list-style-type: none"> 1. java.util.PriorityQueue<E> class 2. insert 3. delete 4. time and space analysis [C4] [C6] 	<p>Resources: Litvin-AB Ch. 7, LCS Sect. 9.8</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Heap Lab—Investigate the workings and code for heaps • Hospital Triage Program using PriorityQueue—Design, implement & test a complete program using a PriorityQueue to determine the order patients are treated • Written Data Structures Summaries & Comparisons • Written Design Questions—Choosing appropriate data structures and justify choices [C3] [C4] [C5] • Test: Heaps & PriorityQueues
11. APCS Case Study—GridWorld [2 weeks]	<ul style="list-style-type: none"> • Use various ADTs to modify existing implementations • Review of previous topics • Interaction & relationships among multiple classes • Extend a given class using inheritance • Object Oriented Design • Pull it all together with the larger program [C6] [C7] 	<p>Resources: APCS GridWorld materials</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Role Play • Review Ch. 1–4 (APCS-A) • Emphasize Ch. 5 • Written assignments in GridWorld • Labs in GridWorld and additional labs modify & extend the existing code using alternate ADTs and algorithms • Test: GridWorld [C6] [C7]

C3—The course teaches students to design and implement computer-based solutions to problems in a variety of application areas.

C4—The course teaches students to use and implement commonly used algorithms and data structures.

C5—The course teaches students to develop and select appropriate algorithms and data structures to solve problems.

C6—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 Appendices A and B of the *AP Computer Science Course Description*. (Note: Students who study a language other than Java in AP Computer Science must also be taught to use Java, as specified in the AP Java subset.)

C7—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®.

Major Units [Time]	Topics	Resources, Labs & Assessments
12. APCS Exam Review [2 weeks]	<ul style="list-style-type: none"> • Review using current College Board Topic Outline • Review selected topics • Test taking techniques • Practice exam questions <ol style="list-style-type: none"> 1. Practice with Quick Reference 	<p>Resources: Review Books, AP Released Exams</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Many multiple choice and free response assignments
13. Independent Projects [4 weeks]	<ul style="list-style-type: none"> • Projects based on student interest 	<p>Resources: Java Web site, texts</p> <p>Labs and Assessments:</p> <ul style="list-style-type: none"> • Projects of student interest <ol style="list-style-type: none"> 1. Maintain log of work 2. Written evaluation of work and processes used