

# AP Computer Science A Syllabus

## Course Description:

The AP Computer Science course is the Third course in the series of computer science courses offered in Prince Georges County. Because the development of computer programs to solve problems is a skill fundamental to the study of computer science, a large part of the course is built around the development of computer programs or parts of programs that correctly solve a given problem. The course also emphasizes the design issues that make programs understandable, adaptable, and, when appropriate, reusable. At the same time, the development of useful computer programs and classes is used as a context for developing other important concepts in computer science, including the development and analysis of algorithms, the development and use of fundamental data structures, and the study of standard algorithms and typical application. In addition, an understanding of the basic hardware and software components of computer systems and the responsible use of these systems are integral parts of the course. This course will prepare the student for the AP Computer Science A exam.

## Course Resources:

Java Software Solutions for AP Computer Science A, J.Lewis, W.Loftus, and C.Cocking, 2<sup>nd</sup> Edition, 2007, Prentice Hall.

Be Prepared for the AP Computer Science Exam In Java, M. Litvin, 3<sup>rd</sup> Edition, 2008, Skylit Publishing.

AP GridWorld Case Study.

Web sites:

<http://www.allstudentscanlearn.org> – Mrs. White’s website. This website should be checked daily for updates. Homework and assessments will be announced here as well as in class.

<http://java.sun.com/javase/downloads/index/jsp>  
Sun’s website to download latest version of the JAVA compiler.

[www.aw-bc.com/cssupport/LewisLoftusCocking.htm](http://www.aw-bc.com/cssupport/LewisLoftusCocking.htm)  
Source code and case studies provided by the textbook authors.

[www.acm.org/serving/se/code.htm](http://www.acm.org/serving/se/code.htm)  
ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices.

Note: A chart is available at the end of the syllabus to show correlation between the “Computer Science A” column of the Topic Outline in the AP Computer Science Course Description and each unit of this syllabus.

## Grading Policy:

Labs – 10%

Classwork/Homework/Warmups – 20%

Assessments – 40%

Seasonal Packets and Projects – 30%

## **Unit Information:**

### **Unit 1: Introduction to Computer Systems & Ethics**

The student will learn about the components of a computer system, hardware and software. The code of ethics provided by the ACM will be explored. The students will download and install the latest JAVA compiler and the GridWorld Case Study, as well as the support material provided by Lewis,Loftus,Cocking.

### **Unit 2: Key Elements of a Program**

This unit explores the key elements of a program. The use of primitive data types and the use of pre-defined objects are introduced. Students are also introduced to design techniques including pseudocode, flowcharting, and structure charts. Students create their first programs. Students will also work through Part One of the GridWorld Case Study.

### **Unit 3: Flow Control Structures**

This unit covers the two basic flow control structures. Students will learn to use both selection and repetition statements to control the flow through a program. The students will also learn about boolean expressions and truth tables.

### **Unit 4: Writing Classes**

This unit covers how to create a class. Proper method structure and class composition is enforced. This unit also explores interfaces and how to use them to create a class.

### **Unit 5: Arrays**

This unit covers the design and use of one dimensional arrays, including using arrays as parameters, and searching and sorting arrays. The use of the ArrayList class and the use of ListIterator will be introduced. Multidimensional arrays will also be explored.

### **Unit 6: Advanced Classes**

This unit covers the concepts of inheritance, class hierarchies, and polymorphism. Abstract classes will be explored. Polymorphism will be explored using both inheritance and interfaces.

### **Unit 7: Recursion**

This unit covers recursion, including what recursion is and when it should be used. Concepts of recursion in searching and sorting are explored.

### **Unit 8: GridWorld Case Study**

Students will complete parts two, three, and four of the GridWorld Case Study.

### **Unit 9: Exam Preparation**

The students will review materials to prepare for the exam. An exam review book will be used, as well as, previous year's exams.

### **Unit 10: Final Project**

Using previously created classes, the student will create a program to solve a specific situational problem. The students may work in a partnership with another student, or work alone.

Sample projects:

Create the game of 21 using the deck of cards created in Chapter 6 of Lewis,Loftus,Cocking

Create the game of dice poker using the die class provided in Chapter 4 of Lewis,Loftus,Cocking.

## **Additional Information:**

Requirement: Minimum of three hours of computer access a week.

Students will have access to the classroom computer lab during class time. Students will also have access to the school computer lab at least 2 days a week after school for up to an hour. Students will be responsible for keeping a log of lab hours and having it signed by the appropriate supervisor.

## Correlation to AP Topic Outline [c2]

<b>I. Object-Oriented Program Design</b>	
The overall goal for designing a piece of software (a computer program) is to correctly solve the given problem. At the same time, this goal should encompass specifying and designing a program that is understandable, can be adapted to changing circumstances, and has the potential to be reused in whole or in part. The design process needs to be based on a thorough understanding of the problem to be solved.	
<b>A. Program design</b>	
1. Read and understand a problem description, purpose, and goals.	Units 1 and 3
2. Apply data abstraction and encapsulation.	Units 2 and 4
3. Read and understand class specifications and relationships among the classes (“is-a,” “has-a” relationships).	Unit 4 and 6
4. Understand and implement a given class hierarchy.	Unit 4
5. Identify reusable components from existing code using classes and class libraries.	Unit 4
<b>B. Class design</b>	
1. Design and implement a class.	Unit 4
3. Choose appropriate data representation and algorithms.	Units 3 and 4
4. Apply functional decomposition.	Units 4
5. Extend a given class using inheritance.	Units 6
<b>II. Program Implementation</b>	
The overall goals of program implementation parallel those of program design. Classes that fill common needs should be built so that they can be reused easily in other programs. Object-oriented design is an important part of program implementation.	
<b>A. Implementation techniques</b>	
1. Methodology	
a. Object-oriented development	Unit 2
b. Top-down development	Unit 3
c. Encapsulation and information hiding	Unit 4 and 6
d. Procedural abstraction	Unit 4 and 6
<b>B. Programming constructs</b>	
1. Primitive types vs. objects	Unit 2
2. Declaration	
a. Constant declarations	Unit 2
b. Variable declarations	Unit 2
c. Class declarations	Unit 4
d. Interface declarations	Unit 4
e. Method declarations	Unit 4
f. Parameter declarations	Unit 4

3. Console output (System.out.print/println)	Unit 2
4. Control	
a. Methods	Unit 4
b. Sequential	Unit 3
c. Conditional	Unit 3
d. Iteration	Unit 3
e. Recursion	Unit 7
C. Java library classes (included in the A-level (AP Java Subset))	Units 2, 4, 5, and 6

<b>III. Program Analysis</b> The analysis of programs includes examining and testing programs to determine whether they correctly meet their specifications. It also includes the analysis of programs or algorithms in order to understand their time and space requirements when applied to different data sets.	
<b>A. Testing</b>	
1. Test classes and libraries in isolation.	Unit 4
2. Identify boundary cases and generate appropriate test data.	Unit 4
3. Perform integration testing.	Unit 4
<b>B. Debugging</b>	
1. Categorize errors: compile-time, run-time, logic.	Units 1, 2, 4 and 5
2. Identify and correct errors.	Units 1, 2, 4 and 5
3. Employ techniques such as using a debugger, adding extra output statements, or hand-tracing code.	Units 1, 2, 4 and 5
C. Understand and modify existing code	Units 1, 2, 3, 4, 5, 6, 7, 8, and 10
D. Extend existing code using inheritance	Units 6
<b>E. Understand error handling</b>	
1. Understand runtime exceptions.	Units 4
<b>F. Reason about programs</b>	
1. Pre- and post-conditions	Units 4
2. Assertions	Unit 4
<b>G. Analysis of algorithms</b>	
1. Informal comparisons of running times	Unit 5
2. Exact calculation of statement execution counts	Unit 5
<b>H. Numerical representations and limits</b>	
1. Representations of numbers in different bases	Unit 1
2. Limitations of finite representations (e.g., integer bounds, imprecision of floating-point representations, and round-off error)	Units 2 and 3

<b>IV. Standard Data Structures</b> Data structures are used to represent information within a program. Abstraction is an important theme in the development and application of data structures.	
A. Simple data types (int, boolean, double)	Unit #2
B. Classes	Units 2, 4, 6, and 10
C. One-dimensional arrays	Unit 5

<b>V. Standard Algorithms</b> Standard algorithms serve as examples of good solutions to standard problems. Many are intertwined with standard data structures. These algorithms provide examples for analysis of program efficiency.	
A. Operations on A-level data structures previously listed	
1. Traversals	Unit 5
2. Insertions	Unit 5
3. Deletions	Unit 5
B. Searching	
1. Sequential	Unit 5
2. Binary	Unit 5
C. Sorting	
1. Selection	Unit 5
2. Insertion	Unit 5
3. Mergesort	Unit 5

<b>VI. Computing in Context</b> A working knowledge of the major hardware and software components of computer systems is necessary for the study of computer science, as is the awareness of the ethical and social implications of computing systems. These topics need not be covered in detail but should be considered throughout the course.	
A. Major hardware components	
1. Primary and secondary memory	Unit 1
2. Processors	Unit 1
3. Peripherals	Unit 1
B. System software	
1. Language translators/compiler	Unit 1
2. Virtual machines	Unit 1
3. Operating systems	Unit 1
C. Types of systems	
1. Single-user systems	Unit 1
2. Networks	Unit 1
D. Responsible use of computer systems	
1. System reliability	Unit 1
2. Privacy	Unit 1
3. Legal issues and intellectual property	Unit 1
4. Social and ethical ramifications of computer use	Unit 1