AP Computer Science A Syllabus

Course Overview

The purpose of this class is to introduce students to object-oriented programming. This course teaches students to use the standard Java library classes from the AP® Java subset given in Appendices A and B of the AP Computer Science Course Description. Concepts such as classes, objects, inheritance, polymorphism, and code reusability are at the heart of the curriculum. Students work in a lab, where constant feedback and help is provided.

The AP Computer Science course is given to challenge students to develop critical thinking skills with hands on tasks. My philosophy is very similar to the one in the sample syllabus. I am the "guide on the side, not the sage on the stage." I introduce a new concept that encourages student-driven learning via questioning and experimentation. I believe in hands-on learning. During classes, I continually monitor student progress and provide individual help as needed.

Texts

Wu, C. Thomas, *An Introduction to Object-Oriented Programming with Java.* McGraw Hill, 2006.
http://www.mhhe.com

Lewis, Loftus, Cocking, Java Software Solutions for AP Computer Science, Addison Wesley, 2004.
http://www.aw-bc.com/

GridWorld Case Study. The College Board, 2006.

Corica, Brown, Presley, *A Guide To Programming in C++,* Lawrenceville Press*, 1997.*
http://www.lvp.com
Programs from this text are used to teach basic conditional and looping structures only.

Units of Study

Unit 1: Introduction to Computers and Programming Languages

This unit teaches students how to use the Eclipse programming environment and plan their work. This is done by learning proper pseudocode and flowchart structures. Students will learn how to use simple input/output, primitive data types, the String class, arithmetic expressions and random number generation. **The students will cover hardware topics which include main and secondary memory location, computer architecture, storage capacity/units, CPUs, peripherals, LANs, WANs, packets, TCP/IP, DNS and others.**

Sample Student Activities for Unit 1: Practice pseudocode and flowchart, Basic Input/Output, Primitive Data Types, Strings, Expressions, Random Input/Output, Ethics, Privacy, Legal Issues, **Primary and secondary memory, Processors, Peripherals**

Resources:

1• **Lewis, Loftus, Cocking: Chapter 1**

Unit 2: Conditional Statements and Looping Structures

Students learn conditional statements, (if– lse, if– lse if, switch) and looping structures (for, while, do while).

Sample Student Activities for Unit 2: Conditionals, Loops, Code Segments

Resources:

2• Wu: Chapters 4 and 5

3• Corica, Brown, Presley: Chapters 4 and 5

Unit 3: Defining Your Own Classes

Students learn how to define a class with multiple methods and data members. Proper method and class structure is emphasized. They will be able to differentiate between local and instance variables. The students will use value-returning methods. Students will learn about constructors and distinguish between public and private methods.

Sample Student Activities for Unit 4: Creating Classes, Modifying/Extending Classes, Methods, Constructors

Resources:
1• Wu: Chapter Four

Unit 4: Defining Your Own Classes Part 2

Students will learn how to overload methods and constructors and define class methods and variables. They will use the reserved word **this** and describe how arguments are passed to parameters and how objects are returned from methods.

Sample Student Activities for Unit 4: Creating Classes, Modifying/Extending Classes, Methods

Resources:
2• Wu: Chapter Seven

Unit 5: Arrays, Arrays of Objects, ArrayLists

Students will be able to manipulate a collection of data values, using an Array. They will also be able to declare and use an Array of Objects and ArrayLists.

Sample Student Activities for Unit 5: Comparing programs using Arrays, Arrays of Objects, ArrayLists. Address book program. Write a method for searching and Array. Traversals, Insertions, Deletions.

Resources:
• Wu: Chapter 10

Unit 6: Strings and the API

Students will use the API to write programs using string manipulation. They should be able to tell the difference between equality and equivalence testing for String objects.

Sample Student Activities for Unit 6: String reversal, String word count, String substitutions, Substrings, Upper/Lower case

Resources:
• Wu: Chapter 9

Unit 7: Inheritance and Polymorphism

Students will learn how to write programs that are easily extended and modifiable by applying polymorphism in program design. Define reusable classes based on inheritance and abstract classes and abstract methods. Differentiate the abstract classes and Java interface. Protected.

Sample Student Activities for Unit 6: Extended the Address book program, Banking program, Assest tracking program.

Resources:

• Wu: Chapter 13

Unit 8: Sorting and Searching

Binary search. Sorting algorithms (Insertion Sort, Selection Sort, and Merge Sort).

1Sample Student Activities: Understand the algorithms behind the following searching/sorting techniques: bubble, selection, and insertion sorts, sequential search and binary search.

Resources:

• Wu: Chapter 11

Unit 9: AP GridWorld Case Study

AP GridWorld Case Study.

Sample Student Activities

• GridWorld Role Play Code Walk-Through, GridWorld
Project 1 – given work with bug variations based on Part 2 of GridWorld Case Study, GridWorld
Project 2 - Design your own class based on material in Part 3 of GridWorld Case Study, GridWorld
Project 3 - Design your own Critters based on material in Part 4 of GridWorld Case Study
GridWorld Case Study Practice and Review

Resources:

1• AP GridWorld Case Study

Unit 10: Input/Output

This unit teaches students how to use input and output streams. Topics include try and catch, exceptions, standard I/O, and reading from and writing to text files.

Resources:
1• Wu: Chapter 12

Unit 11: Recursive Algorithms

The students will write recursive algorithms for mathematical functions and non-numerical operations. The students will be able to decide when to use recursion and when not to.

Resources:
2• Wu: Chapter 15
3

Unit 12: Review for the AP Exam

Students will review for the AP exam.

Resources:
1• AP GridWorld Case Study
2• Wu: Book
3• Handouts

Unit 13: Final Project

After the exam, the students will write a game or application incorporating all the units. Graphics will be incorporated into the game if time allows.

Correlation to AP Topic Outline

I. Object-Oriented Program Design 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.	
A. Program design	
1. Read and understands a problem description, purpose, and goals.	All Units
2. Apply data abstraction and encapsulation.	Units 1-4
3. Read and understand class specifications and relationships among the classes (“is-a,” “has-a” relationships).	Unit 3,4
4. Understand and implement a given class hierarchy.	Unit 3,4
5. Identify reusable components from existing code using classes and class libraries.	Unit 3,4
B. Class design	
1. Design and implement a class.	Unit 3,4
3. Choose appropriate data representation and algorithms.	Units

	2,3,4
4. Apply functional decomposition.	Units 3,4
5. Extend a given class using inheritance.	Units 3,4
II. Program Implementation 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.	
A. Implementation techniques	
1. Methodology	
a. Object-oriented development	Unit 3,4
b. Top-down development	Unit 2,3,4
c. Encapsulation and information hiding	Unit 3,4
d. Procedural abstraction	Unit 7
B. Programming constructs	Unit 2,3,4
1. Primitive types vs. objects	Unit 2,3,4
2. Declaration	
a. Constant declarations	Unit 3,4
b. Variable declarations	Unit 3,4
c. Class declarations	Unit 3,4
d. Interface declarations	Unit 7
e. Method declarations	Unit 3,4
f. Parameter declarations	Unit 3,4
3. Console output (System.out.print/println)	Unit 1
4. Control	
a. Methods	Unit 3,4
b. Sequential	Unit 2
c. Conditional	Unit 2
d. Iteration	Unit 2
e. Recursion	Unit 11

C. Java library classes (included in the A-level (AP Java Subset)	Units 6, 3, 4
III. Program Analysis 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.	
A. Testing	
1. Test classes and libraries in isolation.	Unit 3,4
2. Identify boundary cases and generate appropriate test data.	Unit 3-13
3. Perform integration testing.	Unit 3-13
B. Debugging	
1. Categorize errors: compile-time, run-time, logic.	All Units
2. Identify and correct errors.	All Units
3. Employ techniques such as using a debugger, adding extra output statements, or hand-tracing code.	Units 4-13
C. Understand and modify existing code	Units 4-13
D. Extend existing code using inheritance	Units 7-13
E. Understand error handling	
1. Understand runtime exceptions.	Units 3,4
F. Reason about programs	
1. Pre- and post-conditions	Unit 3,4
2. Assertions	Unit 4
G. Analysis of algorithms	
1. Informal comparisons of running times	Unit 5,8,11
2. Exact calculation of statement execution counts	Unit 5, 8, 11
H. Numerical representations and limits	
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, 3, 4

IV. Standard Data Structures 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 1
B. Classes	Units 3,4
C. One-dimensional arrays	Unit 5
V. Standard Algorithms 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 5,8
2. Binary	Unit 5,8
C. Sorting	
1. Selection	Unit 8
2. Insertion	Unit 8
3. Mergesort	Unit 8
VI. Computing in Context 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/compilers	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