Curricular Requirements	**Page(s)**
CR1: The course teaches students to design and implement computer-based solutions to problems	2,3,4,5,6,7,8,9
CR2a The course teaches students to use and implement commonly used algorithms.	7,9
CR2b The course teaches students to use commonly used data structures.	5
CR3 The course teaches students to select appropriate algorithms and data structures to solve problems	6,7,8,9
CR4 The course teaches students to code fluently in an object-oriented paradigm using the programming language java.	8,9
CR5 The course teaches students to use elements of the Java library from the AP java subset in Appendix A of the AP computer Science A Course Description	4,5
CR6 The course includes a structured lab component compromised of a minimum of 20 hours of hands-on lab experiences.	2
CR7 The course teaches students to recognize the ethical and social implications of computer use.	3

Course Overview

AP® Computer Science A is a provides students with a background in programming that will provide the students with a working understanding of the java language and basic data structures and algorithms. These skills are vital in todays work world and college environment regardless of field. This course will use the 3 suggested AP labs and numerous other programming examples to enforce the core computer science essentials. Students will be comfortable writing java code and will be encouraged to enhance programs for robustness and write programs to solve problems related to other coursework/school activities. The biggest take away form any programming course is developing an ability to break down a problem and with a given set of tools, solve a problem. This skill, even when not programming, is a vital tool in solving any future problems.

Major Texts

Lambert, Kenneth, and Martin Osborne. *Fundamentals of Java: AP Computer Science Essentials.*

Supplemental Texts

Horstman, Cay. *Big Java*. Hoboken, N.J. : Wiley. 2012

Roberts, Eric. *Karel the Robot Learns Java*. Department of Computer Science Stanford University, 2005

Unit 1 (Weeks 0-3)

Topics
- Objects
- Classes
- Conditionals
- Looping

Objectives
- Give Karel simple commands to execute a task
- Learn about classes and how to utilize methods
- Understand the basic loops structures and conditional statements

Assessments: [CR1]
- Write methods using Karel's limited abilities to perform more advanced tasks
- Utilize the different environments to get Karel to perform tasks using simple commands
- Perform the same tasks and more advanced tasks using methods, conditionals, and loops to show the advantages of methods and how to utilize conditionals
- Use the conditional statements to get Karel to complete a beeper maze with beepers evenly spaced or randomly spaced

Unit 2 (Week 4)
Java and Computer Basics

Topics:
- Computer Basics
- Java Basics
- Using the Eclipse IDE
- Understanding input and output

Objectives
- Understand the parts of a computer and a computer network
- Learn about computer ethics such as copyright, intellectual property, shareware, etc.
- Understand the terminology for a programming environment: IDE, JVM, complier etc.
- Run a simple program in Java using a sample code
- Understand what bits and bytes are and how a computer uses binary (base 2)
- Use Scanner.util for input
- Use output with System.out using print and println
- Learn Java Docs and proper commenting
- Learn proper style techniques to making code easy to read and in standard format: Java Docs, indentation, etc.

Assessments: [CR1]
- Labs: Use inputs to multiply and add
- Labs: Use inputs to calculate area and perimeter of basic shapes

- Format a letter given inputs such as name and address
- Label the parts of the computer
- Write about a topic on computer ethics such as pirating, cyber bullying, illegal downloading, viruses etc.

Strategies:
- Assign group work to write basic input output
- Assign a project on computer ethics. Show them where they can find these policies and then write opinion piece on the policy **[CR7]**
- Utilize the lab to show the students small programs and have them edit programs to get desired outcome

Unit 3 (Week 5)
Defining and initializing Variables, Arithmetic Expressions

Topics:
- Using and understanding variables
- Arithmetic expressions in Java
- Using built in Math functions in java

Objectives:
- Understand the different types of numerical variables in Java such as, float, int, double and their limitations
- Learn to declare and initialize variables
- Learn the java environment and how math expressions are handled
- Learn the Math class and Integer class
- Learn how to properly cast a variable in Java
- Understand common java/programming terminology
- Understand the difference between the = and == and other operators in java

Assessments: [CR1]
- Labs:
    - Write a bank program that requires students to enter depositors information and amount of money deposited
    - Modify the bank program to include calculation of the monthly interest

Strategies:
- Present students with program examples correctly and incorrectly handling different variable types. Modify program lines by casting inputs to the correct data type
- Banks program; enter depositor information and calculate deposits and interest

Unit 4 (Weeks 6-7)
Introduction to Classes and Object Oriented Programming

Topics:
- Creating and using classes

Objectives:
- Understand terminology: constructor, getters, setters, instance variables, encapsulation, information hiding, procedural abstraction
- Understand the difference between public and private access in a class
- Use and comprehend the DecimalFormat class and the Random class
- Write classes from scratch, choosing appropriate data representation
- Understand how to declare a method and declare parameters in that method
- Understand the use of preconditions, postconditions and assertions when designing methods
- Understand the difference between OOP development and top-down development

Assessments: [CR1]
- Labs: Purse class and StampMachine class

Strategies:
- Students are given examples and are asked to design classes
- Design classes abstractly representing daily activities and understand which parts of the activity can be represented as a method
- Give students a description and then have students use the techniques learned to designed

Unit 5 (Weeks 8-12)
Conditionals and looping

Topics:
- if, if-else, while, for

Objectives:
- Understand terminology: control statements, counter, infinite loop, iteration, nested loops, logical operators, truth tables
- Construct loops and conditional statements
- Understand errors with loops and conditionals and give students an understanding of debugging techniques
- Use logical operators to make programs more robust
- Construct truth tables
- Ability to analyze a loop and how many times a loop is executed

Assessments: [CR1][CR5]
Labs:
- Approximate PI using Leibniz's method
- Base Conversion: Convert from base
- 10 to base 2
- Guess My Number game
- Euclidean algorithm program
- Perimeter and area of rectangles using all combinations of certain rang**e**

Strategies:
- Write loops and conditionals
- Use hands on material to show students how a loop is working

Unit 6 (Weeks 13-14)
The String Class

Topics:
- String Class

Objectives:
- Instantiate String objects
- Understand that Strings are immutable
- Use appropriate String methods to solve problems
- Begin to emphasize how a string is made of an array of characters

Assessments:
- Lab: Magpie [CR1]

Strategies
- Work several examples using the substring method
- Write code to traverse through strings to locate certain letters or groups of letters

Unit 7 (Weeks 15-17)
Array List

Topic: [CR2b][CR5]
- Using ArrayList class

Objective:
- Use the ArrayList methods

Assessments: [CR1]
- WordList (2004 AP Computer Science A Exam, Free-Response Question 1, AP Central®)

Strategies:
- Stress the difference between add and set
- Draw pictures of the ArrayList after add, set, and remove have been performed
- Use diagrams to understand positioning of elements in an ArrayList

Unit 8 (Week 18)
Arrays

Topics: [CR2b]
- Declaring and initializing arrays
- Manipulating arrays with loops
- Creating parallel arrays

Objectives:
- Understand terminology: array, element, index, logical size, physical size, parallel arrays
- Declare one-dimensional arrays in Java
- Use initializer lists when declaring arrays
- Manipulate arrays using loops and array indices
- Use the physical and logical size of an array together to guarantee they do not go beyond the bounds of their array by identifying the boundary cases and using test data to verify results
- Understand how parallel arrays can be useful when processing certain types of data
- Work with arrays of primitive data types as well as arrays of objects while understanding the difference between the two types of data
- Understand when to choose an array to represent data instead of an ArrayList **[CR3]**

Assessments:
- Lab:
    - For one-dimensional arrays, read in numbers and place each one in an even, odd, and/or negative list [CR1]

Strategies:
- Students need practice manipulating loops that work with arrays
- Students also need to be reminded about the indexing of arrays beginning at zero

Unit 9 (Week 19)
Two-dimensional Arrays

Topics:
- Using 2-D arrays
- Introduction to inheritance and interfaces
- Class diagrams

Objectives:
- Understanding that 2-D arrays are stored as arrays of arrays
- Understand the meaning of row-major order
- Traversing all and part of a two-dimensional arrays
- Using nested loops to manipulate objects in a two-dimensional array

Assessments:
- Lab: [CR1]
    - Picture and Picture lab activities 1-9
    - Picture lab extensions: steganography and chromakey

Strategies:
- Focus on the order in which Java stores the elements of a two-dimensional array in the computer's memory.
- Learn how to write code that corresponds to a class diagram and learn how to draw a class diagram that describes code.

Unit 10 (Weeks 20-21)
Searching and Sorting Arrays

Topic: [CR2a]
- Bubble, Selection, Insertion sorts
- Sequential and Binary Searches

Objectives:
- Write a method for searching an array
- Perform insertions and deletions at given positions in arrays
- Trace through sorting and searching algorithms and understand time constraints of each **[CR3]**
- Understand the algorithm behind each of the following searching and sorting techniques: bubble, selection, and insertion sorts, sequential search and binary search
- Understand Big-O notation and how to select the proper sort for different task base don how the data is represented **[CR3]**
- Identify reusable components from existing code using classes and class libraries
- Given different scenarios, students should be able to choose the most appropriate sort or search **[CR3]**

Assessments:
- Lab:
    - Students make their own "utility" class that includes all of these sorts and searched [**CR1]**
    - Make sure the lab is able to handle different types of data types for the sort

Strategies:
- Students need practice tracing through sorts and searches and determining the runtime of each
- Students also do well with a worksheet that addresses the efficiency of each of the
- strategies they have learned, efficiency for a sorted versus unsorted list, and "best," "worst," and "average" efficiency
- Go over Big-O notations
- Examine http://www.sorting-algorithms.com to show a visual model of each sorting algorithm

Unit 11 (Weeks 22-24)
Elevens Lab

Topics:
- Game design and development
- Experimenting with a large program
- Using classes
- Modifying classes
- Inheritance

Objectives:
- Design a class that models a deck of cards
- Analyze and discuss the efficiency of shuffling algorithms **[CR3]**
- Extend an abstract Board class
- Design a class that models a deck of cards
- Analyze and discuss the efficiency of shuffling algorithms **[CR3]**
- Extend an abstract Board class

Assessments:
- Lab: Elevens

Strategies:
- Be familiar with all the classes and interfaces discussed
- Focus on the how the classes are related to one another and the reasons for preferring one algorithm over another **[CR3][CR4]**

Unit 12 (Weeks 25-27)
More on Classes, Inheritance, Interfaces

Topics:
- Classes
- Inheritance
- Abstract classes
- Interfaces

Objectives:
- Demonstrate inheritance by extending a class
- Understand polymorphism and know when it is appropriate to override methods in a super class
- Create and extend an abstract class
- Create and extend a class given class specifications with the relationships among the classes described
- Implement an interface

Assessments: [CR1]
- Create an abstract Shape class
- Pet Parade (2004 AP Computer Science A Exam: Free-Response Question 2, on AP Central)

Strategies: [CR4]
- Draw pictures of the inheritance hierarchy

Unit 13 (Weeks 28-29)

Topic:
- Inheritance

Objective:
- Use inheritance to extend the Critter class by making new types of Employees

Assessments:
- Exercises from the text

Strategies:
- Have fun with this chapter
- Allow the students to be creative after working through the exercises and analysis
- Create different kinds Employees [CR4]

Unit 14 (Weeks 30-31)
Recursion (and Merge Sort)

Topics:
- Recursion
- Merge Sort [CR2a]

Objectives:
- Create a recursive method to solve a problem
- Understand the difference between recursive and iterative solutions to a problem [CR3]
- Understand and use the Merge Sort
- Understand how to calculate the informal runtime of merge sort and compare it's running time to the other sorts already learned [CR3]

Assessments: [CR1]
- Factorial program
- Rewrite loop programs with recursion

Strategies:
- Ask, "What is returned by this method?"

Unit 15 **(Weeks 32-36)**
Review

Topics:
- Review AP Computer Science A topics

Objective:
- Prepare for the AP Computer Science A Exam by reviewing material and taking practice exams

Assessments:
- Practice exams

Teaching Strategies:
Like learning any language being immersed in the environment is the best way to progress. This is also true of programming and I plan to have students spend most of their time in the

computer lab. In the lab the students will be given problems and be presented with new materials in the java environment. In addition to the computer lab students will work together on being able to visualize a problem and begin to dissect into the necessary part to solve the problem. Problem sets and programs will often have additional topics or enhancements to allow for all students to have a challenge while finishing assignments.