AP Computer Science – AP Syllabus

Teacher Information:

Teacher: Mr. J. Pudaloff
Website: **www.mrpudaloff.com**
School Phone: (248) 823 - 2900

Office Hours: 2nd Hour + arranged
Email: jpudaloff@troy.k12.mi.us
Twitter: @jpudaloff

Textbooks/Supplementary Materials:

- Horstmann, Cay. *Big Java: Early Objects.* Hoboken, NJ. Wiley & Sons, 2014. www.wiley.com/college/horstmann and http://horstmann.com/codecheck
- Cook, Charles E. *Blue Pelican Java*. Refugio, TX: Charles E. Cook, 2010.
- Armstrong, Stacey. A+ Computer Science: Computer Science Curriculum Solutions. http://apluscompsci.com , 2013. Practice site: http://www.practice.apluscompsci.com/
- https://repl.it/ , 2017 Neoreason, Inc.
- Ericson, Barbara Java Review for the AP CS A Exam. http://interactivepython.org/runestone/static/JavaReview/index.html . January, 2014
- Parlante, Nick. Stanford University: *codingbat.com:* `http://codingbat.com/java`
- Microsoft: *Code Hunt*, Java: https://www.codehunt.com/
- University of Washington: *Practice It!:* http://practiceit.cs.washington.edu/practiceit/ *and Code Step By Step:* *https://www.codestepbystep.com/*
- The College Board's Computer Science A Course Description
- Current magazine and Internet articles discussing ethical and social issues related to computer use.
- Additional resources and materials are used and provided throughout the course
- As time permits additional resources and materials are introduced

Course Objectives:

- Understand terminology: CPU, system and application software, primary and secondary memory. Understand how all the different parts of the computer work together
- Understand and apply the main principles of object-oriented software design and programming: classes and objects, constructors, methods, instance and static variables, inheritance, class hierarchies, and polymorphism
- Learn to code fluently in Java in a well-structured fashion and in good style; learn to pay attention to code clarity and documentation

- Learn to use Java library packages and classes
- Understand the concept of an algorithm; implement algorithms in Java using conditional and iterative control structures and recursion.
- Learn to select appropriate algorithms and data structures to solve a given problem.
- Compare efficiency of alternative solutions to a given problem.
- Learn common searching and sorting algorithms: Sequential Search and Binary Search; Selection Sort, Insertion Sort, and Mergesort
- Understand one- and two-dimensional arrays, the List interface, and the ArrayList class, and use them appropriately in programming projects
- Acquire skills in designing object-oriented software solutions to problems from various application areas
- Discuss ethical and social issues related to the use of computers

Software:

The software to be used in this course is Sun Microsystems JDK 1.6 (or newer) and various IDEs including BlueJ and JGrasp. All are freeware. Additional and alternative software is also available if desired.

Suggested Materials:

You should have a folder, notebook, paper and a writing utensil. A 3 ring binder is highly suggested as there are a lot of handouts. Computer/Internet access outside of class (home, library, after school in the lab) can be highly beneficial and is suggested. ***If this is an issue please see the instructor to make alternative arrangements.***

- NOTE: *No supplies are mandatory; all assignments are constructed using materials the school can provide.*

Assessment and Evaluation:

Quarter grades are determined based upon points earned in each of the following weighted categories:

1) Assessments	90%
2) Practice	10%

Grading weighting is tentative and is subject to the instructor's discretion

A+	98-100	B+	88-89	C+	78-79	D+	68-69
A	93-96	B	83-86	C	73-76	D	63-66
A-	90-92	B-	80-82	C-	70-72	D-	60-62
		E	BELOW 60%				

- All students are required to take a final exam at the end of the first (fall) semester
- 1st Semester grade are calculated based on your two quarter grades for a total of 90%. The semester exam will be 10% of your final grade (fall semester).

Remember: Modifications to the weight assignments may be made during the school year, especially in the event that it is beneficial to the student.

Please note that class participation and homework are only *10%* of your cumulative grade. The vast majority of the grade that you will earn in this course will be a reflection of your ability to demonstrate mastery of the content. Please see the quiz policy to understand how reassessments are administered in this course.

AP Computer Science Assessment Policy

During this year, you will be learning information related to many topics (standards) in computer science. A tentative list of standards is posted on Schoology. In this course, a non-traditional criterion-based system is used for assessments. Rather than using a point system to earn a grade for a quiz that may cover many topics, you will be scored on individual standards. You may re-attempt standards provided that you have documented an effort to engage in additional learning (e.g., tutoring, additional practice, quiz corrections, etc.) and scheduled a reassessment with Mr. Pudaloff. **You may not take a reassessment on the same day as the additional learning.** The deadline to re-attempt an assessment on a learning goal is the start of the unit after the subsequent one. For instance, you may retake a quiz from unit 2 any time during unit 3 and before the start of unit 4 unless otherwise noted. If your score on a reassessment of a learning goal exceeds your previous score, the newer and higher score will completely replace the lower score in the gradebook.

Assessment (Standards) grading scale

The material for the course is broken down into several different standards (see list), which will each be scored from 0-5 according to the following scale:

Score	Meaning
10.0	**Expert:** complete mastery of the standard with technical accuracy and demonstrate extensions of my knowledge.
9.0	**Nearly Expert:** mastery of the standard but has not been able to do so with complete accuracy.
8.0	**Proficient:** understanding of the concept, but unable to independently implement an accurate solution. Still something missing about this concept or my solution
7.0	**Developing:** somewhat vague understanding of the concept, but unable to make significant progress towards demonstrating mastery without outside support. Possibly able to accomplish simpler versions of the Proficient tasks.
6.0	**Emerging:** can partially accomplish some of the tasks, but may need extensive help. Needs notes in order to enter into a conversation or solution.
5.0	**Struggling**: Possible evidence of some understanding but no accomplishment of any of the task
4.0	**Nonexistent**: No evidence of understanding

Conference Time

Students are able to receive extra help after school and at lunch. In addition the computer lab will be made available whenever possible. Please schedule a time that is acceptable in advance to ensure availability. Parents may contact me by email or Schoology message, which are the best methods. Most inquiries are answered promptly and all will receive a response within 24 hours. **Email/Schoology are preferred and by far the most effective methods of communication.**

Make-up Work- All make-up work is the responsibility of the student to get, do, and turn in. It is your responsibility to arrange for a make-up quiz or test. One additional day is given for make-up work for each day of excused absence.

Students should check the syllabus, calendar, and Schoology frequently for deadlines and to be aware of what to expect next. Deadlines are subject to change. The best way to start on a path to success is to read and understand your syllabus and refer to it as needed. Once you have read this syllabus to completion please email me a picture of a dinosaur.

- **Electronic devices, such as iPods, lap tops, cell phones, are useable at the instructor's discretion.**

*** These rules are subject to change. All district and school rules will be followed and enforced.***

Consequences For Violations Include At Least:

- **1st Offense –** Conference with student
- **2nd Offense –** Phone call home, detention, and conference with student
- **3rd and Subsequent Offense –** Referral to Assistant Principal

Academic Integrity

This includes cheating and plagiarism. This is very important in computer programming. Students are encouraged to help each other and share thoughts and ideas, but should **never copy code from another student or give solutions or code to another student, this is considered cheating**. In the case of cheating by copying another student's work both students will be subject to punishment. Both plagiarism and cheating will result in a grade of zero and may be subject to further action.

FAQs

Q: Can I get help from another student?
A*: You absolutely can and are encouraged to get help from others. This does not include copying code from another student or that student telling you what to type. It can include others telling you possible problems or mistakes in your code and making general suggestions.*

Q: Can I help another student?
A: *You absolutely can and are encouraged to do so. Helping others actually deepens your understanding and will benefit you also. Seeing alternate approaches or thinking can be a major source of learning and it will make your problem solving skills better. You of course should not give other students solutions but you can point out logic or syntax errors and make suggestions.*

Q: Where else can I get help?
A: *Besides your fellow students and the instructor, there are many places to get help. There are several electronic textbooks available at* mrpudaloff.com *and also hard copies of textbooks available. Also on* mrpudaloff.com *you will find notes and videos for every topic that include examples. Check the supplementary materials for extra places to practice and resources. Of course there is always good old fashioned Google and YouTube which always have thousands of results for every topic. If you are still stuck see the instructor for further suggestions.*

EXTRA HELP

- **Please do not wait** **to get extra help** when you do not understand the material or need further clarification. Remember that sometimes 5 or 10 minutes of individual help makes a big difference in understanding and learning the concepts that are misunderstood. Please ***do not wait*** to get extra help! Rather, talk to me as soon as you do not understand. I check my e-mail at least daily. Email is the best and easiest way to contact me and get a prompt response

- **My e-mail address:**
    - **jpudaloff@troy.k12.mi.us (school)**

Topic Outline 1 (Concurrent with Topic Outline 2):

Topic	Time	Essential Goals
1. Introduction to Hardware, Software, and the Internet – Binary/Hexadecimal/Octal Numbers	1 week	• *To learn about computers and programming* • *To learn how hardware interfaces with software?* • *To learn how data represented internally*
2. An Introduction to Software Development	0.75 weeks	• *To learn how modern software is developed* • *Describe and algorithm in pseudocode* • *To learn what a compiler is*
3. Java Classes, Objects, and Events: A Preview	0.75 weeks	• *Become familiar with your programming environment* • *Compile and run your first Java program* • *Understand the concepts of classes and objects* • *Be able to call methods*
4. Java Syntax and Style	0.5 weeks	• *To learn what a compile time error is* • *To learn what a run time error is* • *Program structure and organization*
5. Data Types, Variables and Arithmetic	1.5 weeks	• *Learn about variables* • *Integer vs. floating point numbers* • *To learn about primitive data* • *Be able to perform arithmetic operations?*
6. Boolean Expressions and `if-else` Statements	2 weeks	• *Be able to write and evaluate boolean expressions* • *Be able to use boolean operators* • *How programs evaluate data and make decisions* • *Validate user input*
7. Methods, Constructors, and Fields	3 weeks	• *Become familiar with the process of implementing classes and write classes* • *To be able to implement and test simple methods* • *To understand the purpose and use of constructors and implement them*

Topic	Time	Essential Goals
		• *To understand and be able to access instance variables and local variables* • *To become familiar with javadoc and the Java API*
8. Strings	2 weeks	• *Be able to create and use Strings* • *Be able to use String methods including parameters and return types* • *Be able to manipulate Strings*
9. Arrays	2 weeks	• *Be able to collect elements using arrays* • *Learn and use common algorithms for processing and searching arrays* • *Be able to work with two dimensional arrays*
10. Iterative Statements: `while`, `for`, `do-while`	2.5 weeks	• *Be able to implement the while, for, and do loops* • *To learn and be able to use common loop algorithms* • *To understand and use nested loops* • *Be able to implement programs that read and process data sets* • *To learn and use the debugger* • *To use the enhanced for loop for traversing arrays*
11. Recursion	1.5 weeks	• *Be able to "think recursively"* • *Be able to implement and use recursive helper methods* • *Understand the relationship between recursion and iteration* • *Understand when recursion is useful and how it relates to program efficiency*
12. Searching, Sorting and Other Array Algorithms	0.75 weeks	• *Study and understand several sorting and searching algorithms* • *Appreciate that algorithms for the same task vary widely in performance* • *Estimate and compare the performance of algorithms*
13. Inheritance / Polymorphism	3.5 weeks	• *Understand and use the concept of inheritance* • *Implement subclasses that inherit and override superclass methods* • *Understand the concept of polymorphism* • *Become familiar with the common superclass Object and its methods*
14. Abstract Classes	1 week	• *Understand, write, and use Abstract classes* • *Understand when Abstract classes are useful and desired*
15. Interfaces, `Lists`, and `ArrayLists`	2.5 weeks	• *Be able to declare and use interface types* • *To appreciate how interfaces can be used to*

Topic	Time	Essential Goals
		decouple classes • *Understand the List interface* • *Be able to collect elements using ArrayLists* • *Be able to use the enhanced for loop to traversing ArrayLists*
16. Case Studies (throughout)	3 weeks	• *To become familiar with large projects* • *Be able to use documentation* • *Be able to modify and analyze existing code* • *Be able to implement subclasses of classes*
17. AP Exam Review	Remaining time before AP Test	• *Prepare for AP Test*
Optional Topics (Time Permitting)		
18. Streams and Files	Remaining class time	• *Be able to read and write text files* • *To throw and catch exceptions*
19. Graphics		• *Understand the basics of graphics in Java* • *To be able to draw and manipulate simple graphics objects*
20. GUI Components and Events		• *Be able to use layout managers to arrange user-interface components in a container* • *Be able to use text components to capture and display text in a graphical application* • *Become familiar with and use common user-interface components, such as radio buttons, check boxes, and menus* • *Browse the Java documentation effectively*

<u>Topic Outline 2 (Concurrent with Topic Outline 1):</u>

TIME	TOPICS
3 days	**Unit 0A – Computer Science, Computer Lab, and Objects Introduction**

AP Topics – Test classes and libraries in isolation; Identify and correct errors : compiletime, 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.

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 a large program, modifying existing code of a large program, and expanding and extending existing code.

TIME	**TOPICS**
Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Hello World, others. Assessments : Labs, Quizzes, and Tests(m/c)	
2 days	**Unit 0B - Computers**
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. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Slides, Worksheets, etc. Labs : BlueJ (basics and if and loop introduction), others. Assessments : Quizzes and Tests(m/c)	
1 week	**Units A and B – Output and Variables**
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. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Draw a shape using ASCII characters, others. Assessments : Labs, Quizzes, and Tests(m/c)	
1 week	**Units C and 1 – Input and Methods**
AP Topics – Variable declarations; Console output; Java library classes; Simple data types(int, boolean, double); Classes; Method declarations, Class declarations; Parameter declarations. **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.	

TIME	TOPICS
Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Create a class that draws some shape of your choosing Assessments : Labs, Quizzes, and Tests(m/c)	
1 week	**Units 2 and 3 – Classes, OOP, Math Operations**
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. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Numerous labs that require students create mathematical expressions in code Assessments : Labs, Quizzes, and Tests(m/c)	
2 days	**Unit 4 - Strings and OOP**
AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Add strings together, find letters in a string, count letters in a string, change letters Assessments : Labs, Quizzes, and Tests(m/c)	
8 days	**Units 5-7 – Conditionals – If, If else, If else if,**

TIME	**TOPICS**
	Switch Case

AP Topics – Conditional; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.

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(`>`,`<`,`>=`,`<=`,`!`).

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

Readings : Labs, Slides, Worksheets, etc.

Labs : Odd and Even, Find Biggest and Smallest, others.

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

7 days	**Units 8-9 – Iteration – For Loop and While Loop**

AP Topics – Iteration; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.

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. Students will learn when to use Integer.MAX_VALUE and Integer.MIN_VALUE.

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

Readings : Labs, Slides, Worksheets, etc.

Labs : GCF, Reverse Numbers and Strings, Find Biggest and Smallest

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

4 days	**Unit 10 - Boolean Logic and Boolean Laws**

AP Topics – Boolean; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.

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.

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

Readings : Labs, Slides, Worksheets, etc.

TIME	TOPICS
Lab : Random Number Guessing, Password Checking Assessments : Labs, Quizzes, and Tests(m/c)	
4 days	**Unit 11 - Iteration – Nested Loops**
AP Topics – Iteration; Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Triangle Output with letters, others. Assessments : Labs, Quizzes, and Tests(m/c)	
1 day	**Units 12-13 – Chopping Strings and File Input (Optional Topics)**
AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Primes, Biggest Numbers, Smallest Numbers, Strings Assessments : Labs, Quizzes, and Tests(m/c)	
2 weeks	**Units 14-15 - One dimensional arrays [!!Critical Topic!!]**
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. **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.	

TIME	TOPICS

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

Readings : Labs, Slides, Worksheets, etc.

Labs : Count a particular number, Histograms, Biggest, Smallest, Next Value

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

3 weeks	**Unit 16 - `ArrayList` [!!Critical Topic!!]**

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.

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 about the `java.util.List` interface. Students gain experience working with a large program, modifying existing code of a large program, and expanding and extending existing code.

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

Readings : Labs, Slides, Worksheets, etc.

Labs : Use Old Free Response Questions as Labs, others.

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

4 days	**Unit 17 – References / Parameters**

AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations.

Student Objectives – Students will learn more about references and parameter passing. Students will learn the differences between passing primitives and references as parameters.

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

Readings : Labs, Slides, Worksheets, etc.

Labs : Array of Pick Your Animal, Use Old Free Response Questions as Labs

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

3 days	**Unit 18 - Interfaces / OOP**

AP Topics – Object Oriented development; Read and understand a problem description, purpose, and goals; Class design; Method declarations; Parameter declarations; Class declarations; Interface declarations.

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. Students gain experience working with a large

TIME	TOPICS
program, modifying existing code of a large program, and expanding and extending existing code. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Sort by Criteria, Use Comparable to Sort Labs : TBD Assessments : Labs, Quizzes, and Tests(m/c and free response)	
3 days	**Unit 19 – Array of References [!!Critical Topic!!]**
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. **Student Objectives –** Students will learn more about storing references in arrays. Students will learn the difference between arrays of primitives and arrays of references. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Array of Pick Your Animal, Use Old Free Response Questions as Labs Assessments : Labs, Quizzes, and Tests(m/c and free resposne)	
END OF SEMESTER ONE	
2 weeks	**Unit 20 – Inheritance [!!Critical Topic!!]**
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. **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. Students will learn how to use static variables. Students gain experience working with a large program, modifying existing code of a large program, and expanding and extending existing code. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets	

TIME	TOPICS
Readings : Labs, Slides, Worksheets, etc. Labs : Make a Game using inheritance and lots of objects – Tic Tac Toe Labs : TBD Assessments : Labs, Quizzes, and Tests(m/c and free response)	
2 weeks	**Unit 21 - Abstract Classes**
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. **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, when to use an abstract class, and when to use static variables. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Build a Game, others. Assessments : Labs, Quizzes, and Tests(m/c and free response)	
3 weeks	**Unit 22 – Matrices [!!Critical Topic!!]**
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. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs: Sort the Matrix, Count a Value in the Matrix, Use Old FR Questions as Labs Assessments : Labs, Quizzes, and Tests(m/c and free response)	
2 weeks	**Unit 23 - Recursion**
AP Topics – Recursion; Object Oriented development; Read and understand a problem	

TIME	**TOPICS**
description, purpose, and goals; Class design; Class declarations. **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 gain experience working with a large program, modifying existing code of a large program, and expanding and extending existing code. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Create a Blob Remover, Blob Counter, Use Old FR Questions as Labs Labs : TBD Assessments : Labs, Quizzes, and Tests(m/c and free response)	
1 week	**Unit 24 - Advanced Sorting and Searching / Comparable**
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. **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. Guided Practice : Topic discussion, Example program analysis and modification, Worksheets Readings : Labs, Slides, Worksheets, etc. Labs : Review all sorting algorithms, Sort by Criteria using Comparable Assessments : Labs, Quizzes, and Tests(m/c and free response)	
8 weeks	**AP Review Time**
AP Topics : Arrays, ArrayList, Inheritance, Labs. Guided Practice : Past year's free response and multiple choice questions Guided Practice : Slides, labs, etc. Readings : Past year's free response and multiple choice questions	

TIME	**TOPICS**
Readings : Review book units	
End of Semester Two	

Computer Science 1-AP Skills Checklist

	• *To learn about computers and programming*
	• *To learn how hardware interfaces with software?*
	• *To learn how data represented internally*
	• *To learn how modern software is developed*
	• *Describe and algorithm in pseudocode*
	• *To learn what a compiler is*
	• *Become familiar with your programming environment*
	• *Compile and run your first Java program*
	• *Understand the concepts of classes and objects*
	• *Be able to call methods*
	• *To learn what a compile time error is*
	• *To learn what a run time error is*
	• *Program structure and organization*
	• *Learn about variables*
	• *Integer vs. floating point numbers*
	• *To learn about primitive data*
	• *Be able to perform arithmetic operations?*
	• *Be able to write and evaluate boolean expressions*
	• *Be able to use boolean operators*
	• *How programs evaluate data and make decisions*
	• *Validate user input*
	• *Become familiar with the process of implementing classes and write classes*
	• *To be able to implement and test simple methods*
	• *To understand the purpose and use of constructors and implement them*
	• *To understand and be able to access instance variables and local variables*
	• *To become familiar with javadoc and the Java API*
	• *Be able to create and use Strings*
	• *Be able to use String methods including parameters and return types*
	• *Be able to manipulate Strings*
	• *Be able to collect elements using arrays*
	• *Learn and use common algorithms for processing and searching arrays*
	• *Be able to work with two dimensional arrays*
	• *Be able to implement the while, for, and do loops*
	• *To learn and be able to use common loop algorithms*
	• *To understand and use nested loops*
	• *Be able to implement programs that read and process data sets*
	• *To learn and use the debugger*
	• *To use the enhanced for loop for traversing arrays*
	• *Be able to "think recursively"*
	• *Be able to implement and use recursive helper methods*

	• *Understand the relationship between recursion and iteration*
	• *Understand when recursion is useful and how it relates to program efficiency*
	• *Study and understand several sorting and searching algorithms*
	• *Appreciate that algorithms for the same task vary widely in performance*
	• *Estimate and compare the performance of algorithms*
	• *Understand and use the concept of inheritance*
	• *Implement subclasses that inherit and override superclass methods*
	• *Understand the concept of polymorphism*
	• *Become familiar with the common superclass Object and its methods*
	• *Understand, write, and use Abstract classes*
	• *Understand when Abstract classes are useful and desired*
	• *Be able to declare and use interface types*
	• *To appreciate how interfaces can be used to decouple classes*
	• *Understand the List interface*
	• *Be able to collect elements using ArrayLists*
	• *Be able to use the enhanced for loop to traversing ArrayLists*
	• *Prepare for AP Test*