AP Computer Science A

Syllabus – 2017-2018 School Year

Tya Chuanromanee, IMS Academy

**Description:** An introductory course to computer science topics, focusing on learning a programming language (Java), problem-solving using algorithms, identifying and utilizing good design practices, learning about applications of algorithms and problem solving skills, and understanding the ethical and social impact of computer use.

**Texts:** Anderson, Julie, and Franceschi, Herve J. *Java Illuminated: An Active Learning Approach (4th Edition)*. Jones & Bartlett Learning, 2014. ISBN: 1284045315

Ethics Readings: <http://jodypaul.com/sweng.html>

**Prerequisites:** Algebra I or equivalent

**Credits:** 1 credit

**Course Overview and Teaching Method**

This course is equivalent to a first-semester, first year course in computer science. In this class, you will learn an object-oriented programming language from scratch. No previous programming knowledge is assumed. From knowing the basics of code, we will learn to apply these coding skills to solve problems of increasing difficulty and scale. Each week we will build upon topics learned in the preceding weeks, so it is important to be able to know and understand the basics. By the time you finish the course, you will be ready to take the AP Computer Science A test.

Lab is a very crucial part of this course. We will spend at least 20 hours doing lab in this course, starting from day 1. Combined with lectures, this gives you a good overview of the applications of what you are learning. In addition to in-class lab time, there will be projects and homework to work on by yourself to be turned in.

The student is expected to do their reading before each class and come to class with questions. Lectures and labs may be conducted in person or via Skype.

**Course Schedule**

­­This is an overview of the main topics addressed by the course as well as the readings schedule. The course is expected to take 30 weeks. The actual coverage may vary from this outline, depending on student performance or other circumstances. There is no guarantee that the instructor will follow this course outline word-for-word in that order. As such, please consider this course outline only as a guideline.

1. **Week 1:** Introduction to the course, getting started with Java (The Very Basics)
   1. Pseudocode, Variables and types, procedural abstraction, operators
   2. Reading: 2.1-2.3.3, 2.3.5-2.3.8
2. **Week 2:** Introduction to classes and objects
   1. Objects, methods, encapsulation
   2. Reading: 3.1-3.4
3. **Week 3:** The Java Class Library
   1. Packages, import, basic classes and how to use them
   2. Reading: 3.6-3.16
   3. Start Project 1
4. **Week 4:** Logical operators and conditions
   1. Equality, logical, and relational operators, if statements
   2. Reading: 5.1-5.5
5. **Week 5:** Conditionals, cont.; Testing; Value Comparisons
   1. Dangling else, test plans, conditions, comparing floats and Strings
   2. Reading: 5.6-5.11
6. **Week 6:** Basic looping
   1. Switch/case, while loops, File class
   2. Reading: 6.1-6.3.2
   3. Start Project 2
7. **Week 7:** Looping Techniques
   1. Accumulation, counting, averages, max/min
   2. Reading: 6.4-6.5
8. **Week 8:** More While Loops
   1. Loop conditions, testing, do/while
   2. Reading: 6.6-6.9
9. **Week 9:** For Loops and Writing Methods
   1. For loops, local and flag variables, nested loops, writing methods, parameters
   2. Reading: 6.10-6.11
   3. Start Project 3
10. **Week 10:** Writing Classes
    1. Defining a class, writing class methods (constructors and accessors), overloading
    2. Reading: 7.1-7.4
11. **Week 11:** Writing Class Methods, cont.
    1. Mutator methods, encapsulation, data manipulation methods
    2. Reading: 7.5-7.6
12. **Week 12:** More Class Methods
    1. Object references, implicit parameters, overriding, static keyword, UML
    2. Reading: 7.9-7.11
    3. Start Project 4
13. **Week 13:** Arrays
    1. One-dimensional arrays and accessing elements
    2. Reading: 8.1-8.3.1
14. **Week 14:** Applications of Arrays
    1. Summing, min/max, copying, changing array size, comparing arrays, clients and access, command line args
    2. Reading: 8.3.1-8.3.7, 8.5
15. **Week 15:** Search and Sort
    1. Sequential search of unsorted array, selection and insertion sort, sorting objects
    2. Reading: 8.6.1-8.6.3
16. **Week 16:** Midterm
    1. No reading, just review
17. **Week 17:** Search, cont.
    1. Sequential search of sorted array, binary search, call by value
    2. Reading: 8.6.5-8.6.6
    3. Start Project 5
18. **Week 18:** Hierarchies, Inheritance
    1. Hierarchies, super/subclasses, inheritance, overriding, inheritance, protected access modifier
    2. Reading: 10.1-10.2.3
19. **Week 19:** Inheritance, cont., Polymorphism
    1. Adding specialization, overriding, abstract classes and methods, polymorphism
    2. Reading: 10.2.4-10.6
    3. Start Project 6
20. **Week 20:** Sorting, cont.
    1. Selection, Bubble, Merge sort
    2. No reading
21. **Week 21:** Arrays, Recursion
    1. Multidimensional arrays, intro to walls and mirrors, into to recursion, functional decomposition
    2. No reading
22. **Week 22:** Recursion
    1. Recursive helper methods, recurrence relations, box trace, applications of recursion
    2. No reading
23. **Week 23:** Recursion, cont., Abstract Data Types
    1. Recursion and efficiency, ADT, information hiding
    2. No reading
24. **Week 24:** Interfaces and Exceptions
    1. Interfaces, interfaces vs abstract classes, basic exceptions, information hiding vs encapsulation
    2. No reading
25. **Week 25:** Using Interfaces and ArrayList
    1. Using the List interface, basics of generics, ArrayList class basics
    2. No reading
26. **Week 26:** Unsigned Numbers and Conversion
    1. Decimals, binary numbers, hexadecimals, octal
    2. Reading TBA
27. **Week 27:** Elements of Design
    1. Bottom-up vs top-down design, modularity, code reusability
    2. Reading TBA
28. **Week 28:** Computing in Context
    1. System reliability, privacy, legal issues and intellectual property, social and ethical ramifications of computer use
    2. Reading TBA
29. **Week 29:** Preparing for the AP Exam
    1. Basics of the exam, preparation tips
    2. Reading TBA
30. **Week 30:** Final Exam

**Assignments:**

Hands-on learning is a big part of computer science. Because of this, there are many assignments in this class.

There will be 6 projects. These projects are expected to take a few weeks to be completed. It is best to not procrastinate when working on these projects, since they require some forethought and design. Earlier projects may take a shorter period of time to complete. The instructor will give students enough information and time for the projects to be completed before the deadlines.

There will be homework assigned periodically throughout the course. The homework will not require writing the code from scratch. Instead, it may deal with solving problems or implementing a part of a pre-written class or code.

**Course Objectives:**

A student who receives credit for this course must be able to:

* Design, implement, and analyze solutions to problems
* Use and implement commonly used algorithms
* Develop and select appropriate algorithms and data structures to solve new problems
* Write solutions fluently in an object-oriented paradigm
* Write, run, test, and debug solutions in the Java programming language, utilizing standard Java library classes and interfaces from the AP Java subset
* Read and understand programs consisting of several classes and interacting objects
* Read and understand a description of the design and development process leading to such a program
* Understand the ethical and social implications of computer use

**Supplementary Materials:**

These materials are not required, but will aid your learning and understanding of the course material.

* [Java Standard Edition (SE) Online Documentation](https://docs.oracle.com/javase/8/docs/api/)
* [jGRASP Online Documentation](http://www.jgrasp.org/tutorials200/jGRASP_02_Getting_Started.pdf)

**Grading Scale:**

This grading scale is used to convert numerical grades into letter grades.

|  |  |  |  |
| --- | --- | --- | --- |
| A | 92-100 | C | 72-74 |
| A- | 88-91 | C- | 70-71 |
| B+ | 85-87 | D+ | 69 |
| B | 81-84 | D | 68 |
| B- | 78-80 | F | 0-67 |
| C+ | 75-78 |  |  |

**Grade Composition:**

60% - Projects (6 x 10% each)

10% - Homework

15% - Midterm

15% - Final