Mapping of Computer Science A to Think Java

This document is an extended version of the *Topic Outline* from the *AP Computer Science A Course Description*, © 2014 The College Board. Additions are shown in **bold green text**. The original version is here: https://apstudent.collegeboard.org/apcourse/ap-computer-science-a

---

Following is an outline of the major topics considered for the AP Computer Science A Exam. This outline is intended to define the scope of the course, but not the sequence.

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, and can be adapted to changing circumstances. The design process needs to be based on a thorough understanding of the problem to be solved.

A.	Program and Class Design		
	1.	Problem analysis	**Section 11.8-9, 14.6**
	2.	Data abstraction and encapsulation	**Section 11.1, 12.1**
	3.	Class specifications,	**Chapter 11**
		interface specifications,	**n/a**
		relationships (“is-a,” “has-a”),	**Section 14.7**
		and extension using inheritance	**Section 14.3**
	4.	Code reuse	**Section 7.4, 14.5**
	5.	Data representation and algorithms	**Chapter 12-14**
	6.	Functional decomposition	**Section 6.3, 13.2**

II. Program Implementation

Part of the problem-solving process is the statement of solutions in a precise form that invites review and analysis. The implementation of solutions in the Java programming language reinforces concepts, allows potential solutions to be tested, and encourages discussion of solutions and alternatives.

A.	Implementation techniques		
	1.	Top-down	**Section 13.2, 14.5**
	2.	Bottom-up	**Section 14.6**
	3.	Object-oriented	**Section 10.5, 14.7**
	4.	Encapsulation and information hiding	**Section 11.1, 12.5**
	5.	Procedural abstraction	**Section 7.3-4, 12.2**
B.	Programming constructs		
	1.	Primitive types vs. reference types	**Chapter 8-11**
	2.	Declaration	
		a. Constants	**Section 3.5, 7.2**
		b. Variables	**Chapter 2**
		c. Methods and parameters	**Section 4.3, 4.5-6**
		d. Classes	**Chapter 11-14**
		e. Interfaces	**n/a**

3.	Text output using System.out.print and println	**Section 1.4-5**
4.	Control	
	a. Method call	**Section 4.1-2**
	b. Sequential execution	**Section 4.4**
	c. Conditional execution	**Section 5.3-4**
	d. Iteration	**Section 7.1, 7.5**
	e. Recursion	**Section 5.8-10, 6.7**
5.	Expression evaluation	
	a. Numeric expressions	**Section 2.5, 2.9**
	b. String expressions	**Section 2.8, 9.3-6**
	c. Boolean expressions, short-circuit, De Morgan	**Section 5.1-3**

C. Java library classes and interfaces included in the AP Java Subset

Object	**Section 14.3**
Integer	**Section 9.8**
Double	**Section 9.8**
String	**Chapter 9**
Math	**Chapter 4-7**
List	**n/a**
ArrayList	**Section 14.1-2**

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.	Development of appropriate test cases,	**Section 6.2-3**
	including boundary cases	**n/a**
2.	Unit testing	**Section A.7**
3.	Integration testing	**Section A.4**

B. Debugging

1.	Error categories: compile-time, run-time, logic	**Section 2.10**
2.	Error identification and correction	**Appendix C**
3.	Techniques such as using a debugger,	**Section A.6**
	adding extra output statements,	**Section 12.9**
	or hand-tracing code.	**Section 4.7, 5.9, 8.2, 10.6**

C. Runtime exceptions

ArithmeticException	**Section 2.10, C.2**
IllegalArgumentException	**n/a**
IndexOutOfBoundsException	**Section 8.2, 9.3, C.2**
NullPointerException	**Section 10.7, 12.6, C.2**

D. Program correctness

1. Pre- and post-conditions	**n/a**
2. Assertions	**Section A.7**

E. Algorithm Analysis
   1. Statement execution counts — **Section 8.8-9, 12.7**
   2. Informal running time comparison — **Section 13.4**

F. Numerical representations of integers
   1. Non-negative integers in different bases — **Section 5.10**
   2. Implications of finite integer bounds — **Section 4.1**

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. Primitive data types (int, boolean, double) — **Section 2.1, 2.6, 5.1**
B. Strings — **Chapter 2, 9**
C. Classes — **Section 11.1**
D. Lists — **Section 14.1-2**
E. Arrays (1-dimensional and 2-dimensional) — **Chapter 8**

V. Standard Operations and 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 data structures
   1. Traversals — **Section 8.6, 9.3**
   2. Insertions — **n/a**
   3. Deletions — **n/a**

B. Searching
   1. Sequential — **Section 12.7**
   2. Binary — **Section 12.8**

C. Sorting
   1. Selection — **Section 13.3**
   2. Insertion — **Section 13.9**
   3. Mergesort — **Section 13.4-7**

VI. Computing in Context

An awareness of the ethical and social implications of computing systems is necessary for the study of computer science. These topics need not be covered in detail, but should be considered throughout the course.

A. System reliability — **n/a**
B. Privacy — **n/a**
C. Legal issues and intellectual property — **n/a**
D. Social and ethical ramifications of computer use — **n/a**

VII. Other Topics in the AP Java Subset

The AP Java subset is intended to outline the features of Java that may appear on the AP Computer Science A Exam. The AP Java subset is NOT intended as an overall prescription for computer science courses — the subset itself will need to be supplemented in order to address all topics in a typical introductory curriculum.

A.	**Comments**	**Section 1.4, 4.9**
B.	**Concatenation**	**Section 2.8**
C.	**Escape Sequences**	**Section 1.6**
D.	**Numeric casts**	**Section 3.7**
E.	**Object Comparison**	**Section 11.7, 12.4**
F.	**Packages (import)**	**Section 3.2**
G.	**Visibility**	**Section 11.4, 11.9**

VIII. Topics in Think Java but not in AP

The following sections of Think Java present material that is not part of the AP Java Subset and/or Topic Outline above.

*** State/stack diagrams**	**Section 2.3, 4.7**
*** Console input (Scanner)**	**Section 3.2, 3.10**
*** Formatted output (printf)**	**Section 3.6**
*** Using the Javadoc tool**	**Section 4.9**
*** Parsing/validating input**	**Section 5.7, 9.8**
*** Do-while, break, continue**	**Section 7.6-7**
*** Using java.util.Arrays**	**Section 8.3-4**
*** Command-line arguments**	**Section 9.9**
*** UML class diagrams**	**Section 10.9, 14.7**
*** Java library source code**	**Section 10.10**
*** Explicit use of “this”**	**Chapters 11-13**
*** DrJava, Checkstyle, JUnit**	**Appendix A**
*** Java 2D graphics library**	**Appendix B**