TOPIC OUTLINE

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
   2. Data abstraction and encapsulation
   3. Class specifications, interface specifications, relationships ("is-a," "has-a"), and extension using inheritance
   4. Code reuse
   5. Data representation and algorithms
   6. Functional decomposition

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
   2. Bottom-up
   3. Object-oriented
   4. Encapsulation and information hiding
   5. Procedural abstraction

B. Programming constructs
   1. Primitive types vs. reference types
   2. Declaration
      a. Constants
      b. Variables
      c. Methods and parameters
      d. Classes
      e. Interfaces
   3. Text output using `System.out.print` and `System.out.println`
   4. Control
      a. Method call
      b. Sequential execution
      c. Conditional execution
      d. Iteration
      e. Recursion

    5. Expression evaluation
        a. Numeric expressions
        b. String expressions
        c. Boolean expressions, short-circuit evaluation, De Morgan's law
C. Java library classes and interfaces included in the AP Java Subset

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, including boundary cases
    2. Unit testing
    3. Integration testing
B. Debugging
    1. Error categories: compile-time, run-time, logic
    2. Error identification and correction
    3. Techniques such as using a debugger, adding extra output statements, or hand-tracing code.
C. Runtime exceptions
D. Program correctness
    1. Pre- and post-conditions
    2. Assertions
E. Algorithm Analysis
    1. Statement execution counts
    2. Informal running time comparison
F. Numerical representations of integers
    1. Representations of non-negative integers in different bases
    2. Implications of finite integer bounds

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)
B. Strings
C. Classes
D. Lists
E. Arrays (1-dimensional and 2-dimensional)

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
   2. Insertions
   3. Deletions

B. Searching
   1. Sequential
   2. Binary

C. Sorting
   1. Selection
   2. Insertion
   3. Mergesort

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
B. Privacy
C. Legal issues and intellectual property
D. Social and ethical ramifications of computer use