AP Computer Science A		
Standards	**Fall Semester**	
College Board	Topics Covered	Number of Days
	AP Computer Science A *introduces students to computer science through programming. Fundamental topics in this course include the design of solutions to problems, the use of data structures to organize large sets of data, the development and implementation of algorithms to process data and discover new information, the analysis of potential solutions, and the ethical and social implications of computing systems. The course emphasizes object-oriented programming and design using the Java programming language.*	
	Semester 1	
2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 4.B Identify errors in program code. 1.A Determine an appropriate program design to solve a problem or accomplish a task (not assessed). 1.B Determine code that would be used to complete code segments. 2.A Apply the meaning of specific operators. 5.A Describe the behavior of a given segment of program code. 5.B Explain why a code segment will not compile or work as intended.	**Chapter 1: Introduction to Computers:** Hardware Terminology, Main Memory, Auxiliary Memory, Drives, Writing Algorithms Using Pseudocode, Programming Language Code, The Compilation Process for Non-Java Programs, Object Code, Portability, Java Virtual Machine, The Compilation Process for Java Programs, History of Java, Computer Ethics	10
5.A Describe the behavior of a given segment of program code. 1.C Determine code that would be used to interact with completed program code. 3.A Write program code to create objects of a class and call methods. 2.C Determine the result or output based on the statement execution order in a code segment containing method calls.	Chapter 2: Algorithms and Design 2.1. Introduction 2.2. Output 2.3. Variables 2.4. Operators and Assignment Statements 2.5. Input 2.6. Flow of Control and Flowcharts 2.7. if Statements 2.8. Loops 2.9. Loop Termination Techniques 2.10. Nested Looping 2.11. Tracing 2.12. Problem Solving: Other Pseudocode Formats and an Asset Management Example	10

2.A Apply the meaning of specific operators. 2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements. 4.A Use test-cases to find errors or validate results. 3.A Write program code to create objects of a class and call methods.	Chapter 3 : Java Basics 3.1. Introduction 3.2. “I Have a Dream” Program 3.3. Comments and Readability 3.4. The Class Heading 3.5. The main Method’s Heading 3.6. Braces 3.7. System.out.println 3.8. Compilation and Execution 3.9. Identifi ers 3.10. Variables 3.11. Assignment Statements 3.12. Initialization Statements 3.13. Numeric Data Types— int , long , float , double 3.14. Constants 3.15. Arithmetic Operators 3.16. Expression Evaluation and Operator Precedence 3.17. More Operators: Increment, Decrement, and Compound Assignment 3.18. Tracing 3.19. Type Casting 3.20. char Type and Escape Sequences 3.21. Primitive Variables Versus Reference Variables 3.22. Strings 3.23. Input—the Scanner Class 3.24. Simple File Input for Repetitive Testing During Program Development	10
1.A, 1.B, 2.A, 2.B, 3.A, 3.C	Six Week Exam	1
1.B Determine code that would be used to complete code segments. 2.B Determine the result or output based on statement execution order in a code segment without method calls (other than output). 2.D Determine the number of times a code segment will execute. 3.C Write program code to satisfy method specifications using expressions, conditional statements, and iterative statements. 4.C Determine if two or more code segments yield equivalent results. 5.C Explain how the result of program code changes, given a change to the initial code.	Chapter 4: Control Statements 4.1. Introduction 4.2. Conditions and Boolean Values 4.3. if Statements 4.4. && Logical Operator 4.5. | | Logical Operator 4.6. ! Logical Operator 4.7. switch Statement 4.8. while Loop 4.9. do Loop 4.10. for Loop 4.11. Solving the Problem of Which Loop to Use 4.12. Nested Loops 4.13. boolean Variables 4.14. Input Validation	10

1.A 3.B 5.A 5.B 5.D	Chapter 5: Using Prebuilt Methods 5.1. Introduction 1 63 5.2. The API Library 1 64 5.3. Math Class 1 67 5.4. Wrapper Classes for Primitive Types 1 72 5.5. Character Class 176 5.6. String Methods 178 5.7. Formatted Output with the printf Method 184 5.8. Problem Solving with Random Numbers 5.9. GUI Track: Drawing Images, Lines, Rectangles, and Ovals in Java Applets 5.10. GUI Track: Decorating an Image by Covering It With a Tinted Pane	10
3.D 4.B	Chapter 6: Object-Oriented Programming 6.1. Introduction 6.2. Object-Oriented Programming Overview 6.3. First OOP Class 6.4. Driver Class 6.5. Calling Object, this Reference 6.6. Instance Variables 6.7. Tracing an OOP Program 6.8. UML Class Diagrams 6.9. Local Variables 6.10. The return Statement 6.11. Argument Passing 6.12. Specialized Methods—Accessors, Mutators, and Boolean Methods 6.13. Problem Solving with Simulation	9
3.D 4.B1.A 3.B 5.A 5.B 5.D	Six Week Exam	1
2.C 2.D 3.D 5.C	Chapter 7: Object-Oriented Programming—Additional Details 7.1. Introduction 7.2. Object Creation—A Detailed Analysis 7.3. Assigning a Reference 7.4. Testing Objects for Equality 7.5. Passing References as Arguments 7.6. Method-Call Chaining 7.7. Overloaded Methods 7.8. Constructors 7.9. Overloaded Constructors 7.10. Class Variables 7.11. Class Methods 7.12. Named Constants 7.13. Problem Solving with Multiple Driven Classes	10
1.B 2.B 2.D 3.E	Chapter 8: Software Engineering 8.1. Introduction 8.2. Coding-Style Conventions 8.3. Documentation for Outsiders 8.4. Helper Methods Confirming Proofs 8.5. Encapsulation (with Instance Variables and Local Variables) 8.6. Recognizing the User's Point of View 8.7. Design Philosophy 8.8. Top-Down Design 8.9. Bottom-Up Design 8.10. Case-Based Design 8.11. Iterative Enhancement 8.12. Merging the Driver Method into the Driven Class 8.13. Accessing Instance Variables Without Using this 8.14. Writing Your Own Utility Class 8.15. Problem Solving with the API Calendar Class 8.16. GUI Track: Problem Solving with CRC Cards	9

1.B 2.B 2.D 3.E1.A 3.B 5.A 5.B 5.D2.C 2.D 3. D, 5.C	E1 Exam	1