AP Computer Science A
Course Syllabus and Planning Guide
(2021-2022)

Curricular Requirements

Curricular Requirements		Page(s)
CR1	The teacher and students have access to a college-level computer science textbook, in print or electronic format.	3
CR2	The course provides opportunities to deepen student understanding of the required content outlined in each of the units described in the AP Course and Exam Description.	4
CR3	The course provides opportunities to deepen student understanding of the Big Ideas.	5, 6, 7, 8, 9, 10, 11, 12, 13, 14
CR4	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 1: Program Design and Algorithm Development.	5, 6, 7, 8, 10, 11, 12, 13
CR5	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 2: Code Logic.	5, 6, 7, 8, 9, 10, 11, 12, 14
CR6	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 3: Code Implementation.	6, 7, 8, 9, 10, 11, 12, 13, 14
CR7	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 4: Code Testing.	5, 7, 14
CR8	The course provides opportunities for students to develop the skills related to Computational Thinking Practice 5: Documentation.	5, 6, 7, 8, 10, 14
CR9	This course provides students with hands-on lab experiences to practice programming through designing and implementing computer-based solutions to problems.	4

Advanced Placement Computer Science A

Computer science embraces problem solving, hardware, algorithms and perspectives that help people utilize computers to solve real-world problems in everyday life. The AP Computer Science A course introduces students to computer science with fundamental topics that include problem solving, design strategies and methodologies, organization of data (data structures), approaches to processing data (algorithms), analysis of potential solutions, and the ethical and social implications of computing. The course emphasizes both object-oriented and imperative problem solving and design. These techniques represent proven approaches for development solutions that can scale up from small, simple problems to large, complex problems.

By the end of this course, students will be able to:

- Design and implement computer-based solutions to problems.
- Use and implement commonly used algorithms and data structures.
- 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
- Read and understand programs consisting of several classes and interacting objects
- Read and understand a description of the design and development process
- Understand the ethical and social implications of computer use.

Project STEM's course is endorsed by the College Board and an authorized AP® Computer Science A course.

Prerequisites

This course requires a strong foundation in English and mathematics. Students must be comfortable with functions and the concepts found in the uses of functional notation. Prior computer programming experience is not required.

Teaching Strategies

The course will consist of video lectures, daily programming exercises, longer coding assignments, and regular quizzes and exams. Students will also participate in online discussion forums. By the end of the course, students will be well prepared to take the AP® Computer Science A exam.

Each lesson listed below includes practice exercises, including shorter coding problems. Well over 20 hours of instructional time is spent in hands-on coding using the course coding exercises, lab assignments and AP labs. Students participate regularly in a moderated discussion forum that provides support for lesson material and also introduces discussions of the ethical implications of programming.

Textbook

The following textbooks are used in the course:

- Online text: AP Computer Science A by Project STEM.
- Textbook: David J Eck's "Introduction to Programming Using Java." (2013).

Course Outline

The course includes the required content organized into the following units based on the AP Course and Exam Description:

- Unit 1: Primitive Types
- Unit 2: Using Objects
- Unit 3: Boolean Expressions and if Statements
- Unit 4: Iteration
- Unit 5: Writing Classes
- Unit 6: Array
- Unit 7: ArrayList
- Unit 8: 2D Array
- Unit 9: Inheritance
- Unit 10: Recursion

To fulfill the 20-hour lab requirement, the course will use the provided College Board labs as suggested in the guides:

- Consumer Review Lab
- Data Lab
- Steganography Lab
- Celebrity Lab

Additionally, the course will provide the following optional labs for additional hands-on lab experience:

- Magpie Lab: this lab implements a chatbot and incorporates String manipulations.
- Elevens Lab: students will apply design and object oriented principles to create and play a solitaire game.
- Picture Lab: students will write methods that modify digital pictures and learn to traverse a 2D array of integers or objects.

Course Sequencing

Unit 1: Primitive Types

This unit introduces students to the basics of programming in Java, focusing on the use of variables and operators for storing and manipulating primitive data.

Duration

- 2 Weeks

Unit Topics

- User input and output
- Variables
- Data types
- Calculations using int and double values
- Modular division
- Numeric casting

Lessons

- Unit 1: Lesson 1 - Output In Java: *This lesson demonstrates the system.out.print and system.out.println commands to print text (**MOD**).*
- Unit 1: Lesson 2 - User Input and Variables: *In this lesson, students learn how to store data in variables to be accessed again later (**VAR**).*
- Unit 1: Lesson 3 - Data Types: *This lesson describes different primitive data types that can be used to store numbers in Java (**VAR**).*
- Unit 1: Lesson 4 - Number Calculations: *This lesson introduces arithmetic operators and shows how to perform calculations in Java (**CON**).*
- Unit 1: Lesson 5 - Modular Division: *This lesson explains how modular divisions can be calculated in Java (**CON**).*
- Unit 1: Lesson 6 - Numeric Casts: *This lesson introduces casting to convert one data type to another (**CON**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will complete a partial line of code (**Skill 1.B**)
- Students will determine the results of printing an expression concatenating two strings (**Skill 2.A**).
- Students will explain why a given line of code is incorrect (**Skill 5.B**)
- Students will find an alternative expression which gives the same results in a code segment (**Skill 4.C**).
- Students will determine the results of running a section of code with multiple unary and assignment operators (**Skill 2.B**).
- Students will design a program to individually print out the digits of a three-digit number (**Skill 1.A**).
- Students will determine the result of casting a number (**Skill 2.A**).

Assignment

- Assignment 1: Movie Ratings: *Students will write code using arithmetic operators and variable assignment correctly to calculate values required by the assignment specification (**Skill 2.A**). They will need to fix any errors which occur in their code before submitting it (**Skill 4.B**).*

Assessments

- Unit 1 Quiz
- Unit 1 Exam

Unit 2 - Using Objects

This unit introduces the use of classes, methods and objects.

Duration

- 4 Weeks

Unit Topics

- Primitive vs. class data types
- Computer memory and storage
- String functions and concatenation
- Constructors, classes and objects
- Void and non-void methods
- Wrapper classes
- The Math class

Lessons

- Unit 2: Lesson 1 - Strings and Class Types: *Through the specific example of the String data type, this lesson explains how reference data is stored in memory (**VAR**).*
- Unit 2: Lesson 2 - Escape characters and String Concatenation. *This lesson explains how to use escape characters in String literals, and more about concatenating String variables with primitives (**VAR**)*
- Unit 2: Lesson 3 - String Functions: *This lesson introduces methods which can be called on objects of the String data type (**VAR**).*
- Unit 2: Lesson 4 - Classes and Objects: *This lesson explains the class-object structure found in Java (**MOD**).*
- Unit 2: Lesson 5 - Using Constructors: *In this lesson students are introduced to new classes they can import, and use constructors for creating objects of these class types (**MOD**).*
- Unit 2: Lesson 6 - Using Methods: *This lesson explains how to call methods on objects in Java, and also introduces API documentation which is used to determine which methods can be used for a variable of a given class (**MOD, VAR**).*
- Unit 2: Lesson 7 - Wrapper Classes: *This lesson discusses wrapper classes and explains the processes of autoboxing and unboxing to convert between wrapper class-type variables and primitives (**VAR**).*
- Unit 2: Lesson 8 - Math Functions: *This lesson introduces the static methods from the Math class, used for further mathematical calculations in Java code (**CON**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will determine the output of a statement with multiple different operators used (**Skill 2.A**).
- Students will determine the result of executing a line of code containing multiple method calls (**Skill 2.C**).
- Students will use methods to take user input, then create objects based upon the input (**Skill 3.A**).
- Students will choose a code segment to change the state of an object which is declared in the question text (**Skill 1.C**)
- Students will write code which creates objects, then call methods to both determine and change the state of the object (**Skill 3.A**).
- Students will explain why a line of code is incorrect (**Skill 5.B**).

Assignment

- Assignment 2: Control Tower: *Students will use the constructors of a new class type to create objects, and call methods on the objects created to both change and access their states (**Skill 3.A**).*

Assessments

- Unit 2 Quiz
- Unit 2 Exam

Unit 3 - Boolean Expressions and If Statements

This unit introduces the concepts that programs use to make decisions using the logic of Boolean expressions, If and Else statements.

Duration

- 3 Weeks

Unit Topics

- If and else statements
- Booleans and truth tables
- Short circuit evaluation and De Morgan's law
- Comparing objects

Lessons

- Unit 3: Lesson 1 - Simple Ifs: *In this lesson, students are shown how to use if statements to introduce conditional logic to their program (**CON**).*
- Unit 3: Lesson 2 - Ifs - Making Decisions: *This lesson introduces more comparison operators which can be used in if statements with primitive values (**CON**).*
- Unit 3: Lesson 3 – Else: *This lesson introduces the else and else if statements to give students greater control over the flow of their programs (**CON**).*
- Unit 3: Lesson 4 - Booleans and Truth Tables: *This lesson introduces the use of Boolean operators to create compound Boolean statements (**CON**).*
- Unit 3: Lesson 5 - Short Circuit Evaluation: *This lesson explains how short-circuit evaluation is used by Java when evaluating Boolean statements (**CON**).*
- Unit 3: Lesson 6 - De Morgan's Law: *In this lesson students learn how to apply De Morgan's law to find equivalent Boolean statements (**CON**).*
- Unit 3: Lesson 7 - Comparing Objects: *This lesson explains how to compare object data in Java using both the '==' operator and the equals method (**CON**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will identify an error in a short segment of code (**Skill 4.B**).
- Students will identify which code segments from a selection are equivalent (**Skill 4.C**).
- Students will complete a partially completed code segment (**Skill 1.B**).
- Students will write code with multiple conditional statements to produce a desired output (**Skill 3.C**).
- Students will describe the details of how a particular statement will be evaluated in a Java program (**Skill 5.A**).
- Students will identify equivalent Boolean statements (**Skill 4.C**).
- Students will evaluate the output from a code segment containing multiple conditionals (**Skill 2.B**).

Assignment

- Assignment 3 - Crack the Code!: *Students will write nested decision statements to control the flow of a program, determining which conditions are checked at which points in the program to produce the correct output (**Skill 2.B**).*

Assessments

- Unit 3 Quiz
- Unit 3 Exam

Unit 4 - Iteration

This unit introduces the idea of *iteration*, repeating a process until certain conditions are met. This unit focuses on while and for loops to create algorithms for Strings and numbers. Unit 4 also contains the first College Board lab assignment.

Duration

- 3 Weeks

Topics

- Algorithms for numbers and Strings
- While, for and nested loops

Lessons

- Unit 4: Lesson 1 - While Loops: *In this lesson, students are introduced to the use of while loops to repeat program code (**CON**).*
- Unit 4: Lesson 1 1/2 - Tracing Code: *This lesson demonstrates how the results of code segments containing iterative statements can be determined by using a trace table (**CON**).*
- Unit 4: Lesson 2 - Algorithms for Numbers: *This lesson introduces standard algorithms which students can code, modify and develop using loops (**CON**).*
- Unit 4: Lesson 3 - The For Loop: *This lesson introduces the for loop, an alternative loop syntax used to repeat code (**CON**).*
- Unit 4: Lesson 4 - Algorithms for Strings: *This lesson focuses on the creation, implementation and modification of algorithms which use String traversals (**CON**).*
- Unit 4: Lesson 5 - Nested Loops: *In this lesson, the way in which nested iteration statements can be used is examined (**CON**).*
- Unit 4: Lesson 6 - Algorithm Efficiency: *This lesson introduces the concept of algorithmic efficiency through the use of statement execution counts (**CON**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will explain why a given code segment will not run as intended (**Skill 5.B**).
- Students will explain what is calculated by a given code segment (**Skill 5.A**).
- Students will complete missing code in a loop header (**Skill 1.B**).
- Students will use both selection and iteration to produce a desired output when the main method of a program executes (**Skill 3.C**).
- Students will explain the results of running a code segment (**Skill 5.A**).
- Students will determine the results of running a code segment with nested iteration statements (**Skill 2.B**).
- Students will write nested iteration statements to produce a desired output (**Skill 3.C**).

Assignments

- Assignment 4: String Shortener: *Students will design a program which processes a String, determines properties of this String and prints a modified version of the String (**Skill 1.A**). Students must write iteration statements, decision statements, and String methods to implement this program (**Skill 2.B**).*

Assessments and Labs

- Unit 4 Quiz
- Unit 4 Exam
- Lab: Consumer Review

Unit 5 - Writing Classes

This unit introduces the concepts required for students to write their own classes, which will form the basis for creating more sophisticated programs later in the course. This unit also contains the first of two lessons examining the wider impacts of computer science on society and culture.

Duration

- 4 Weeks

Topics

- Void and Return methods
- Parameters
- Classes
- Constructors
- Static vs. Instance
- Wider Impacts of Computing

Lessons

- Unit 5: Lesson 1 - Void methods: *This lesson explains how code can be written in methods which can then be called from the main code (**MOD**).*
- Unit 5: Lesson 2 - Parameters: *In this lesson, students are shown how to write methods which use parameters to send information with a method call (**MOD**).*
- Unit 5: Lesson 3 - Parameters - Primitive vs. Class: *This lesson explains how parameters of primitive and class-type data are passed to methods when they are called (**MOD**).*
- Unit 5: Lesson 4 - Return Methods: *This lesson introduces the use of return statements to allow methods to send information back when they are called (**MOD**).*
- Unit 5: Lesson 5 - Classes - The Basics: *Through this lesson, students are introduced to the basics of custom class creation, including writing constructors and instance methods/variables (**MOD**).*
- Unit 5: Lesson 6 - Constructors: *This lesson shows in greater depth how constructors can be written to set the initial state of an object, and explains how formal parameters and local variables differ from global variables in scope (**MOD, VAR**).*
- Unit 5: Lesson 7 - Documenting a Class: *In this lesson students learn how to write comments to document a class, including using preconditions and postconditions (**MOD**).*
- Unit 5: Lesson 8 - Static Vs. Instance: *This lesson covers the static modifier which can be used to associate variables or methods to a class rather than with an object (**MOD**).*
- Unit 5: Lesson 9 - Wider Impacts of Computing: *For this lesson, students must conduct research into aspects of the ethical and social implications of computing systems (**IOC**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will determine what is printed by a method that uses selection (**Skill 2.B**).
- Students will write a method which uses selection to meet a specification for what will be printed (**Skill 3.C**).
- Students will determine what is output by a code segment which calls a method (**Skill 2.C**).
- Students will write a method using iteration and selection which returns values according to a specification (**Skill 3.C**).
- Students will create a custom class, which is used to create objects representing vehicles (**Skill 3.B**).

Assignment

- Assignment 5: Fraction: *Students will write the constructors and methods of a class designed to represent fractions (**Skill 3.B**). They will need to write the class methods according to strict specifications as to how the methods should behave for different values of parameters and/or member variables (**Skill 3.C**).*

Assessments

- Unit 5 Quiz
- Unit 5 Exam

Unit 6 – Array

This unit introduces the array data structure which can be used to hold multiple primitive values or object references. Students will learn how to write code which traverses an array to search for or manipulate data in the array.

Duration

- 3 Weeks

Topics

- One-dimensional arrays
- Algorithms for arrays, search algorithms
- Arrays of Strings
- The for-each loop

Lessons

- Unit 6: Lesson 1 - One-Dimensional Arrays: *This lesson introduces the array data type and explains how to access and change the elements of an array (**VAR**).*
- Unit 6: Lesson 2 - Algorithms – Searching: *This lesson demonstrates how an array can be traversed and showcases some standard algorithms which can be used with arrays (**VAR, CON**).*
- Unit 6: Lesson 3 - Arrays of Strings: *This lesson introduces arrays of class data types, focusing on the use of arrays of String objects (**VAR**).*
- Unit 6: Lesson 4 - Algorithms on Arrays: *This lesson introduces algorithms which change the contents of arrays by changing the position of elements as well as adding and removing elements (**CON**).*
- Unit 6: Lesson 5 - The For-Each Loop: *This lesson introduces students to the for-each loop (also called the enhanced for loop) which is used to traverse arrays (**VAR**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will determine the results of executing a block of code with multiple operators working on array elements (**Skill 2.B**).
- Students will determine code needed to complete a partially created code segment (**Skill 1.B**).
- Students will write code which traverses an array and returns information about its contents (**Skill 3.D**).
- Students will describe the effect of changing an existing segment of code (**Skill 5.C**).
- Students will write a method which traverses an array and changes the value of its elements by inserting an element (**Skill 3.D**).
- Students will determine the output of a code segment with multiple method calls (**Skill 2.C**).

Assignment

- Assignment 6: Array Statistics: *Students will write code which creates an array, and then traverses this array to initialize its values and determine certain features (**Skill 3.D**).*

Assessments

- Unit 6 Quiz
- Unit 6 Exam

Unit 7 - ArrayList

This unit introduces the concept of ArrayLists, along with some common search and sort algorithms. This unit also contains the second lesson examining the wider impacts of computer science, with an emphasis on data collection.

Duration

- 3 Weeks

Topics

- ArrayList
- Using the For-Each loop with ArrayLists
- Linear search
- Selection & Insertion sort
- Wider impacts of data collection

Lessons

- Unit 7: Lesson 1 - ArrayList: *This lesson explains what an ArrayList is, and the advantages to using an ArrayList (**VAR**).*
- Unit 7: Lesson 2 - Traversing ArrayLists: *This lesson describes how to traverse ArrayLists using both for-loops and for-each loops, including introducing some algorithms which add or remove items (**VAR, CON**).*
- Unit 7: Lesson 3 - Array Algorithms with ArrayLists: *This lesson demonstrates how standard algorithms developed for arrays can also be applied to work with ArrayList structures (**CON**).*
- Unit 7: Lesson 4 - Linear Search: *This lesson describes the theory of a linear/sequential search algorithm, and also shows how this is implemented for an ArrayList (**CON**).*
- Unit 7: Lesson 5 - Selection Sort: *The selection sort algorithm, used to order data in an array, ArrayList or similar structure, is described in this lesson (**CON**).*
- Unit 7: Lesson 6 - Insertion Sort: *A second algorithm for sorting data, the insertion sort, is introduced in this lesson (**CON**).*
- Unit 7: Lesson 7 - Wider Impacts of Data Collection: *In this lesson, students explore the risks to privacy of storing personal data using computer systems (**IOC**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will write code which creates and then manipulates the contents of an ArrayList (**Skill 3.D**).
- Students will determine the output of a code segment which makes multiple calls to ArrayList and String methods (**Skill 2.C**).
- Students will write a method according to exact specifications given (**Skill 3.C**).
- Students will create a method which implements a selection sort algorithm: manipulating an ArrayList by changing the order of its values (**Skill 3.D**).
- Students will work out the value of a variable by determining the number of times a particular statement runs in a method call (**Skill 2.D**).

Assignment

- Assignment 7: Game Wheel: *Students will write a private method in a partially completed "GameWheel" class which interacts with the other methods and variables of this class as well as a separate class "Slice" which is provided (**Skill 1.C**). This traverses an ArrayList of objects and changes their order (**Skill 3.D**).*

Assessments and Labs

- Unit 7 Quiz
- Unit 7 Exam
- Lab: Data Lab

Unit 8 - 2-D Arrays

This unit introduces the concept of 2-D arrays, along with some algorithms that can be used in conjunction with 2-D arrays.

Duration

- 2 Weeks

Topics

- 2-D Arrays and Algorithms

Lessons

- Unit 8: Lesson 1 - 2-D arrays: *This lesson covers the fundamentals of how data is stored and accessed in a 2-D array (**VAR**).*
- Unit 8: Lesson 2 - 2-D Array Algorithms: *This lesson introduces common algorithms for 2-D arrays, including applying algorithms which were initially defined for 1-D arrays (**CON**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will complete a partially completed code segment by determining a correct condition to use (**Skill 1.B**).
- Students will write a method which is used to create a 2-D array and traverse it to edit the values stored to desired results (**Skill 3.E**).
- Students will trace code which traverses a 2-D array and determine what output is when the code is executed (**Skill 2.B**).

Assignments, Labs and Assessments

- Assignment 8: Battleship: *Students will write methods which traverse a 2-D array, representing a board in a battleship game. The methods will change values in the array as well as returning values to determine features of the board (**Skill 3.E**).*

Assessments and Labs

- Unit 8 Quiz
- Unit 8 Exam
- Lab: Steganography

Unit 9 - Inheritance

This unit introduces the concept of *inheritance,* the ability to have functionality extend from one class to another. The relationships this process creates are examined through the use of class hierarchy diagrams and examples where objects are declared and initialized as different types.

Duration

- 3 Weeks

Topics

- Inheritance
- Inheritance overriding methods
- Is-a and has-a relationships

Lessons

- Unit 9: Lesson 1 – Inheritance – *This lesson introduces inheritance by showing how a class can be extended to create a subclass (**MOD**).*
- Unit 9: Lesson 2 - Inheritance Overriding Methods: *This lesson describes in further detail how methods from a parent class can be overridden in the child class (**MOD**).*
- Unit 9: Lesson 3 - Is-a and Has-a Relationships: *This lesson describes in further detail the relationship between parent and child classes, including the effect of declaring objects of a child type as the parent type (**MOD**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- *Students will create a new class (**Skill 3.B**).*
- *Students will write their code so it interacts with existing code by extending a class for which they are given the source code (**Skill 1.C**).*
- Students will override a method according to specifications by calling a method from the superclass (**Skill 3.C**).
- Students will make decisions about how they would design a class hierarchy for a program (**Skill 1.A**).

Assignment

- Assignment 9: Ultimate Frisbee: *Students will create multiple classes, related by inheritance, to represent different types of people in a sporting scenario (**Skill 3.B**).*

Assessments and Labs

- Unit 9 Quiz
- Unit 9 Exam
- Lab: Celebrity

Unit 10 - Recursion

This unit introduces the concept of *recursion* – defining a process in terms of itself. In practice, this means writing methods which contain calls to themselves to solve a problem.

Duration

- 3 Weeks

Topics

- Recursion
- Recursive functions
- Binary search
- Merge sort

Lessons

- Unit 10: Lesson 1 - Intro to Recursion: *This lesson introduces recursive methods, which are methods that call themselves (**CON**).*
- Unit 10: Lesson 2 - Recursive Functions with Returns: *This lesson demonstrates how values can be returned by recursive methods (**CON**).*
- Unit 10: Lesson 3 - Binary Search: *This lesson explains how a binary search algorithm finds a value in an ordered data set (***CON**).
- Unit 10: Lesson 4 - Merge Sort: *This lesson covers the merge sort algorithm as an example of a recursive sorting algorithm which can be used to put data in order (**CON**).*

This unit's lessons include the following activities that reinforce the course's computational thinking practices:

- Students will determine the output when a recursive method is called (**Skill 2.C**).
- Students will determine how many times a line of code is executed in a certain method call (**Skill 2.D**).
- Students will explain why a given method will not execute correctly (**Skill 5.B**).
- Students will write a recursive method according to a given specification (**Skill 3.C**).
- Students will test their method with a variety of test cases (**Skill 4.A**).

Assignments

- Assignment 10: Anagrams: *Students will carefully test a recursive method by writing a variety of different test cases (**Skill 4.A**).*

Assessments

- Unit 10 Quiz
- Unit 10 Exam

Unit 11 - AP Exam Prep

Students will be given a diagnostic exam and practice AP problems to prepare them to take the AP Computer Science exam. A series of review videos are provided covering the major topics on the AP Exam. The goal of the unit is to allow students to synthesize the material covered throughout the year and review any areas that could use strengthening.

Duration

- 3 Weeks

Topics

- AP Exam Review

Review Lessons

- Programming Fundamentals
- Data Structures
- Logic
- Algorithms
- Object-Oriented Programming
- Recursion
- Software Engineering

Practice Assessments

- Diagnostic Exam
- Released AP Exam Free-Response Questions