AP Computer Science A
Syllabus 2026/27

Course Overview

Why should you study computer science? Computer science is more than computers or code; at its core, computer science is a problem solving skill, essential for your life. Computer science is a dynamic and rapidly growing field of study that has become an integral part of the world that we live in today. By studying computer science, you will not only be able to solve complex, challenging problems, but apply these skills to so many other fields that are only now being developed.

What is the course like? AP Computer Science is a *college level course* that teaches you how to program in the Java computing language. You will learn how to design solutions to problems, use data structures to organize data, develop and implement algorithms, and analyze solutions.

Why Java? By any measure, Java is one of the top computer languages in the world. A person knowledgeable in java can generally pick up other top rated languages such as Python, Javascript, or C# without too much difficulty. Computer science forms the single largest job market with one of the highest pay rates of the science, technology, engineering, and math (STEM) disciplines.

Who Should Take This Class? Traditionally, students with an interest in careers related to business, engineering, computer science, information technology, bioinformatics, genetics, physics, chemistry, or math should take AP computer science. However, in today's world, almost every occupation or endeavor uses some form of computer software...including the fine arts. Some knowledge about how software is designed, created and maintained will provide useful background knowledge.

Course Outcomes

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 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.

Materials

- Charged Laptop from my cart- classwork and homework will be both written and typed; you will be using a laptop every class period

- Writing Utensil and Highlighters- notes will be given out as hard copies and you are expected to follow along and annotate them.
- Three Ring Binder - notes, classwork, and homework will always have a three hole punch for you to keep organized in your binder, by unit.

Grading

- 60% Tests - AP Style Multiple Choice and Free Response Problems (handwritten code) & Unit Projects - AP Level Programming Assignments (typed code)
- 40% Daily Work - Homework, quizzes, activities, programming assignments, etc.

Reassessments

Reassessments are a way to demonstrate understanding at any time after a unit test and before the end of the semester. In order to reassess you must first make sure that you have no missing homework assignments for the unit and then submit test corrections. Your test corrections (written or typed) must meet the following criteria:

- Identify what your mistakes are and explain – This can include understanding of the question, operational errors, use of the wrong method, or something else. "I didn't know how to answer the question" is not an acceptable answer. Your gaps/errors must be explained.
- Indicate how to solve the problem – Explain in a few words the process you will use to solve the problem.
- Work out the test problem(s) showing detail and arriving at a correct solution.

Test corrections *must* be received and reviewed *prior* to reassessing.

You may reassess after receiving your reviewed test corrections. This can be done during Friday Retesting.

Student Responsibility

You must take responsibility for your own learning. Students will be given an agenda at the beginning of each new unit, and a copy will be posted in the Google Classroom. This agenda includes the focus of each day's lesson, as well as assignments to be completed. Students should expect homework to be assigned almost daily, *due the next class period*, unless specified otherwise.

From the GMC student handbook:
A Greer Middle College (GMC) student accepts academic rigor as the main focus of high school, expects to attend college for training or a degree, has reached the maturity and skill level required to begin taking college courses, has reached a developmental level that allows independent work, and seeks the challenge to learn and to grow.

Makeup Work

If a student is absent, he/she will be assigned a grade of zero (0) for all work missed. It is the student's responsibility to obtain and complete the work missed for each absence.

Check the class agenda to see what assignments were missed and what notes/worksheets, if any, need to be picked up. Have a buddy to help you with any notes you missed.

Any assignments that were due on the date(s) of your absence are due immediately upon your return to class.

If you are absent on test day, you will be responsible for making alternative arrangements to take the test during my next scheduled Office Hour (Mondays).

AP Exam Information

The AP Computer Science A Exam is a two-part test.

Section	*Question Type*	*Number of Questions*	*Time Allowed*	*Percent of AP Score*
I	Multiple Choice ● Choices A – D	42 Single Response Questions	90 Minutes	55%
II	Free Response ● Handwritten Code	● Question 1: Methods and Control Structures ● Question 2: Class Design ● Question 3: Data Analysis with ArrayList ● Question 4: 2D Array	90 Minutes	45%

All students are expected to take the AP exam. Students scoring 3 or higher on the AP exam may qualify for a college course exception, dependent upon the college or university. Even if you do not score "well" on the exam, research shows that those students who take an AP exam, in addition to the course, are more likely to obtain a higher level degree.

Help Sessions

If you find that you are confused on any topic, please do not hesitate to ask for help. Do not wait; it will only hurt you in the long run. Although there may not be enough time in class to get assistance, there are other means available to you. Office Hours are on Mondays. Also, there is a wealth of information/lessons online to assist you.

Lab Requirement

You will be asked to join an IDE during the first unit where you will be coding and running your programs. Because of the one-to-one devices you have and the coding assignments that will be assigned each unit, we will meet the College Board's requirement of a minimum of 20 hours of hands-on, structured lab experiences to engage you in individual or group problem solving.

Ethical Use of AI

AI tools like ChatGPT, GitHub Copilot, or CodeWhisperer can be helpful for learning, but they must be used responsibly.

- AI is a learning tool, not a shortcut. It can help you understand, debug, or practice, but it should never replace your own thinking.
- You need to learn the skills yourself. Success on the AP Exam means you must be able to code and solve problems without help from AI.
- Knowing how code works matters more than just getting a solution from a bot.
- Use AI to build deeper understanding, not to copy answers.
- Always check and test AI-generated code because it isn't always correct!

Use AI tools to support your learning, not to do the work for you. You'll be expected to write your own code and explain it clearly.

Finally….

A strong association exists between regular class attendance and successful completion of the class. While it is understood that absences cannot be completely avoided, students are strongly encouraged to be present in school whenever possible, ready and willing to learn.

So, put your best foot forward and let's have a great year!!

Course Sequencing

We will be utilizing the AP Computer Science A CED.

Unit 1: Using Objects and Methods

In this unit, you'll dive into Java programming and learn about variables and classes. These are fundamental concepts that form the basis for everything else we'll do. You'll discover three main types of data and how to create variables to store information. We'll explore how to manipulate these variables using basic operations to reflect real-world situations in our programs. You'll also get familiar with essential Math and String functions that will help you perform calculations and work with text. By the end, you'll be comfortable calling and using different classes to build your programs.

Material Covered	*CED Topics*	*CED Skills*
Lesson 1: Welcome to Java · 1.1.A Represent patterns and algorithms found in everyday life using written language or diagrams. · 1.1.B Explain the code compilation and execution process. · 1.1.C Identify types of programming errors.	1.1	1.A
Lesson 2: Variables and Data Types · 1.2.A Identify the most appropriate data type category for a particular specification. · 1.2.B Develop code to declare variables to store numbers and Boolean values.	1.2	1.A, 2.A
Lesson 3: Expressions and Output · 1.3.A Develop code to generate output and determine the result that would be displayed. · 1.3.B Develop code to utilize string literals and determine the result of using string literals. · 1.3.C Develop code for arithmetic expressions and determine the result of these expressions. · 1.4.A Develop code for assignment statements with expressions and determine the value that is stored in the variable as a result of these statements. · 1.4.B Develop code to read input. · 1.6.A Develop code for assignment statements with compound assignment operators and determine the value that is stored in the variable as a result.	1.3, 1.4, 1.6	2.A, 3.A, 3.D, 4.A

Lesson 4: Casting and Range of Variables · 1.5.A Develop code to cast primitive values to different primitive types in arithmetic expressions and determine the value that is produced as a result. · 1.5.B Describe conditions when an integer expression evaluates to a value out of range. · 1.5.C Describe conditions that limit accuracy of expressions.	1.5	2.A, 4.A
Lesson 5: Creating and Storing Objects · 1.12.A Explain the relationship between a class and an object. · 1.12.B Develop code to declare variables to store reference types. · 1.13.A Identify, using its signature, the correct constructor being called. · 1.13.B Develop code to declare variables of the correct types to hold object references. · 1.13.C Develop code to create an object by calling a constructor. · 1.14.A Develop code to call instance methods and determine the result of these calls.	1.12, 1.13, 1.14	4.A, 2.B, 2.C, 3.C
Lesson 6: Creating and Calling an Object's Methods · 1.8.A Describe the functionality and use of code through comments. · 1.9.A Identify the correct method to call based on documentation and method signatures. · 1.9.B Describe how to call methods.	1.8, 1.9	4.B, 3.C
Lesson 7: String Manipulation · 1.7.A Identify the attributes and behaviors of a class found in the libraries contained in an API.	1.7, 1.15	4.A, 2.C, 3.C, 4.A, 4.B
Lesson 8: Math Class · 1.10.A Develop code to call class methods and determine the result of those calls. · 1.11.A Develop code to write expressions that incorporate calls to built-in mathematical libraries and determine the value that is produced as a result.	1.10, 1.11	2.C, 4.A

Unit 1 Activities and Projects:

- Circuit Activity: Tracing Expressions (Skills: 3.A, 4.A)
- Cut-And-Paste Object Creation Activity (Skills: 2.A, 3.A, 4.A)
- String and Math Methods Station Activity (Skills: 2.A, 3.A, 3.B, 3.C, 4.A)
- PA 1: My First Programs (Skills: 1.A, 2.A)
- PA 2: Creating Objects (Skills: 1.A, 2.A, 2.B)
- PA 3: Strings (Skills: 1.A, 2.A, 2.B, 2.C, 3.A, 4.A)
- PA 4: Math Class (Skills: 1.A, 2.A, 2.B, 2.C, 3.A, 4.A)

Unit 1 Assessments:

- Unit 1 Quiz 1 – MCQs and FRQs covering lessons 1 – 3
- Unit 1 Quiz 2 – MCQS and FRQs covering lessons 4 – 6
- Unit 1 Quiz 3 – MCQs and FRQs covering lessons 7 – 8
- Unit 1 Test – 10 multiple choice questions and 2 free response questions
- Unit 1 Coding Project – Calculator and String Analyzer

Unit 2: Selection and Iteration

Computer programs are built using three main ideas: sequencing, selection, and iteration. You already saw sequencing in Unit 1, and now it's time to focus on the other two. Selection lets your program make decisions using if statements. This helps your code react to different situations, like in games, simulations, or when filtering data. Iteration means repeating actions using loops, like for and while. This is useful when you want the same code to run multiple times. You'll also learn how to write Boolean expressions using comparisons (>, <, ==) and logical operators (&&, ||, !). Plus, you'll practice standard algorithms that use loops, which are patterns you can learn and adapt to solve new problems more easily.

Material Covered	*CED Topics*	*CED Skills*
Lesson 1: Conditional Statements · 2.1.A Represent patterns and algorithms that involve selection and repetition found in everyday life using written language or diagrams. · 2.2.A Develop code to create Boolean expressions with relational operators and determine the result of these expressions. · 2.3.A Develop code to represent branching logical processes by using selection statements and determine the result of these processes.	2.1, 2.2, 2.3	1.A, 2.A, 3.A, 4.A
Lesson 2: Nested Selection · 2.4.A Develop code to represent nested branching logical processes and determine the result of these processes.	2.4	2.A, 4.A
Lesson 3: Compound Boolean Expressions · 2.5.A Develop code to represent compound Boolean expressions and determine the result of these expressions. · 2.6.A Compare equivalent Boolean expressions · 2.6.B Develop code to compare object references using Boolean expressions and determine the result of these expressions.	2.5, 2.6	2.A, 3.A
Lesson 4: while Loops · 2.7.A Identify when an iterative process is required to achieve a desired result. · 2.7.B Develop code to represent iterative processes using while loops and determine the result of these processes.	2.7	2.A, 3.A, 4.B
Lesson 5: Implementing Selection and Iteration Algorithms · 2.9.A Develop code for standard and original algorithms (without data structures) and determine the result of these algorithms.	2.9	2.A, 3.A, 4.A
Lesson 6: for Loops · 2.8.A Develop code to represent iterative processes using for loops and determine the result of these processes.	2.8	2.A, 3.A, 3.D

Lesson 7: Implementing String Algorithms · 2.10.A Develop code for standard and original algorithms that involve strings and determine the result of these algorithms.	2.10	2.C, 3.C, 3.D
Lesson 8: Nested Iteration · 2.11.A Develop code to represent nested iterative processes and determine the result of these processes. · 2.12.A Calculate statement execution counts and informal run-time comparison of iterative statements.	2.11, 2.12	2.A, 3.D, 4.A, 4.B

Unit 2 Activities and Projects:

- Activity: Create Your Own Conditional (Skills: 1.A, 2.A)
- Circuit Tracing Practice: if Statements (Skills: 3.A)
- Circuit Tracing Practice: while Loops (Skills: 3.A)
- Activity: Loop Sorting (Skills: 2.A, 3.A)
- PA 1: Conditionals and Booleans (Skills: 1.A, 2.A, 3.A)
- PA 2: Loops (Skills: 1.A, 2.A, 3.A)
- PA 3: String Algorithms (Skills: 1.A, 2.A, 3.A)

Unit 2 Assessments:

- Quiz 1 – MCQs and FRQs covering lessons 1 – 3
- Quiz 2 – MCQs and FRQs covering lessons 4 – 6
- Quiz 3 – MCQs and FRQs covering lessons 7 and 8
- Unit 2 Test – 10 multiple choice questions and 2 free response questions
- Unit 2 Coding Project – Game of Chance

Unit 3: Class Design

In this unit, you'll take what you've learned so far and use it to create your own custom classes in Java. This means you'll design your own data types to model real-world things (a book, a car, or a student) in your programs. Being able to create classes gives you the freedom to represent your own ideas, not just rely on what's already built into Java. You'll learn how to choose the right characteristics (attributes) and actions (methods) for your classes. We'll also talk about the big picture: how the programs we write can affect people and society, and why it's important to think about the legal and ethical responsibilities of being a programmer.

Material Covered	*CED Topics*	*CED Skills*
Lesson 1: Impacts of Program Design · 3.2.A Explain the social and ethical implications of computing systems	3.2	5.A
Lesson 2: Program Design · 3.1.A Represent the design of a program by using natural language or creating diagrams that indicate the classes in the program and the data and procedural abstractions found in each class by including all attributes and behaviors.	3.1	1.A

Lesson 3: Classes and Constructors · 3.3.A Develop code to designate access and visibility constraints to classes, data, constructors, and methods. · 3.4.A Develop code to declare instance variables for the attributes to be initialized in the body of the constructors of a class.	3.3, 3.4	1.A, 2.B, 3.B
Lesson 4: Accessor and Mutator Methods · 3.5.A Develop code to define behaviors of an object through methods written in a class using primitive values and determine the result of calling these methods.	3.5	2.C, 3.C, 3.D, 4.B
Lesson 5: Class Variables and Methods · 3.7.A Develop code to define behaviors of a class through class methods. · 3.7.B Develop code to declare the class variables that belong to the class.	3.7	2.C, 3.B
Lesson 6: References, Scope, and this · 3.6.A Develop code to define behaviors of an object through methods written in a class using object references and determine the result of calling these methods. · 3.8.A Explain where variables can be used in the code. · 3.9.A Develop code for expressions that are self-referencing and determine the result of these expressions.	3.6, 3.8, 3.9	2.B, 2.C, 3.B, 3.D, 4.A

Unit 3 Activities and Projects:

- Activity: Ethical Design Role Play (Skills: 5.A)
- Activity: Think-Pair-Share the FRQs (Skills: 1.A, 2.A, 2.B, 2.C, 4.A, 4.B)
- PA 1: Creating Objects (Skills: 1.A, 2.A, 2.B, 2.C, 4.A, 4.B)
- PA 2: Static Methods (Skills: 1.A, 2.A, 2.B, 2.C, 4.A, 4.B

Unit 3 Assessments:

- Quiz 1 – MCQs and FRQs covering lessons 1 – 3
- Quiz 2 – MCQs and FRQs covering lessons 4 – 6
- Unit 3 Test – 10 multiple choice questions and 2 free response questions
- Unit 3 Coding Project – Hangman

Unit 4: Data Collections

In this unit, you'll learn how to work with different ways to store lots of data in Java - arrays, ArrayLists, and 2D arrays. Arrays are fixed in size, but ArrayLists can grow and shrink, making it easier to add or remove items. You'll also explore 2D arrays, which are great for organizing data in tables, and learn how to loop through them in different ways. We'll also cover useful algorithms for sorting and searching through data, and you'll get introduced to recursion, which is a cool way of solving problems by breaking them into smaller pieces. Along the way, we'll talk about data privacy and how big data can be used in both helpful and harmful ways.

Material Covered	*CED Topics*	*CED Skills*

Lesson 1: Ethical and Social Issues Around Data Collection · 4.1.A Explain the risks to privacy from collecting and storing personal data on computer systems · 4.1.B Explain the importance of recognizing data quality and potential issues when using a data set. · 4.1.C Identify an appropriate data set to use in order to solve a problem or answer a specific question.	4.1	1.B, 5.A
Lesson 2: Data Sets · 4.2.A Represent patterns and algorithms that involve data sets found in everyday life using written language or diagrams.	4.2	1.B
Lesson 3: One Dimensional Arrays · 4.3.A Develop code used to represent collections of related data using one-dimensional (1D) array objects.	4.3	2.B, 3.B
Lesson 4: Array Traversals · 4.4.A Develop code used to traverse the elements in a 1D array and determine the result of these traversals.	4.4	2.B, 3.B, 3.D
Lesson 5: Implementing Array Algorithms · 4.5.A Develop code for standard and original algorithms for a particular context or specification that involves arrays and determine the result of these algorithms.	4.5	2.B, 3.B, 3.D, 4.A
Lesson 6: Using Text Files · 4.6.A Develop code to read data from a text file.	4.6	2.C, 3.C, 4.B
Lesson 7: Wrapper Classes · 4.7.A Develop code to use Integer and Double objects from their primitive counterparts and determine the result of using these objects.	4.7	2.B
Lesson 8: ArrayList · 4.8.A Develop code for collections of related objects using ArrayList objects and determine the result of calling methods on these objects.	4.8	2.B, 3.B, 3.D
Lesson 9: ArrayList Traversals · 4.9.A Develop code used to traverse the elements of an ArrayList and determine the result	4.9	2.B, 3.D, 4.A
Lesson 10: Implementing ArrayList Algorithms · 4.10.A Develop code for standard and original algorithms for a particular context or specification that involve ArrayList objects and determine the result of these algorithms.	4.10	2.B, 3.B, 3.D, 4.A
Lesson 11: 2D Arrays · 4.11.A Develop code used to represent collections of related data using two-dimensional (2D) array objects.	4.11	2.B, 3.B

Lesson 12: 2D Array Traversals · 4.12.A Develop code used to traverse the elements in a 2D array and determine the result of these traversals.	4.12	2.B, 3.B, 3.D
Lesson 13: Implementing 2D Array Algorithms · 4.13.A Develop code for standard and original algorithms for a particular context or specification that involves 2D arrays and determine the result of these algorithms.	4.13	2.B, 3.B, 3.D, 4.A
Lesson 14: Searching Algorithms · 4.14.A Develop code used for linear search algorithms to search for specific information in a collection and determine the results of executing a search.	4.14	2.B, 3.B, 4.A
Lesson 15: Sorting Algorithms · 4.15.A Determine the result of executing each step of sorting algorithms to sort the elements of a collection.	4.15	3.B, 4.A, 4.B
Lesson 16: Recursion · 4.16.A Determine the result of calling recursive methods.	4.16	3.C, 4.A, 4.B
Lesson 17: Recursive Searching and Sorting · 4.17.A Determine the result of executing recursive algorithms that use strings or collections. · 4.17.B Determine the result of each iteration of a binary search algorithm used to search for information in a collection. · 4.17.C Determine the result of each iteration of the merge sort algorithm when used to sort a collection.	4.17	4.A, 4.B

Unit 4 Activities and Projects:

- Activity: Tracing with Arrays (Skills: 3.A, 3.B)
- Activity: Stations with ArrayLists (Skills: 3.A, 3.B, 4.A)
- Activity: Think-Pair-Share FRQs with ArrayLists (Skills: 1.A, 2.A, 2.C, 4.B)
- Activity: Representing 2D Data (Skills: 1.A, 2.A, 2.B, 4.A)
- Activity: Sorting and Searching (Skills: 1.A, 2.A, 3.C)
- Activity: Think-Pair-Share FRQs with 2D Arrays (Skills: 1.A, 2.A, 2.C, 4.B)
- Activity: Tracing Recursion (Skills: 3.A, 3.C)
- PA 1: Arrays (Skills: 1.A, 2.A)
- PA 2: Reading Text Files (Skills: 1.A, 1.B, 2.A, 2.B)
- PA 3: ArrayLists (Skills: 1.A, 2.A, 2.B)
- PA 4: 2D Arrays (Skills: 1.A, 1.B, 2.A, 2.B)

Unit 4 Assessments:

- Quiz 1 – MCQs and FRQs covering lessons 3 – 5
- Quiz 2 – MCQs and FRQs covering lessons 6 & 7
- Quiz 3 – MCQs and FRQs covering lessons 11 – 13
- Quiz 4 – MCQs and FRQs covering lessons 14 – 17
- Unit 4 Test Part 1 – 10 multiple choice questions and 2 free response questions (Lessons 1 – 10)
- Unit 4 Test Part 2– 10 multiple choice questions and 2 free response questions (Lessons 11 – 17)
- Unit 4 Project 1: Arrays and ArrayLists
- Unit 4 Project 2: 2D Arrays