

# AP Computer Science Principles Summer Packet

---

*Please note: This exam is different than other AP exams you have taken in the past.*

Students enrolled in AP Computer Science Principles will be expected to complete the following over the course of the summer:

**Thoroughly read the course information and description found at:**

<https://apstudents.collegeboard.org/courses/ap-computer-science-principles>

You will need to know how the exam is formatted and what is expected of you.

Download the detailed exam description. It is very important that you know what you must complete before you begin the course. You will be *tested* on what is included in this document:

<https://apcentral.collegeboard.org/media/pdf/ap-csp-student-task-directions.pdf>

Watch the documentary “**Terms and Conditions May Apply**” available online

Cullen Hoback exposes the erosion of online privacy and what information governments and corporations are legally taking from citizens each day.

## ~ Graded Assignments ~

### Code.org Unit 3 Lessons:

- You will need to create an account on [Code.org](https://code.org). Make sure that you use your Pace email and a username which includes your ACTUAL name so that I can identify you.
- Once you create an account, you will need to put in the section code: [SKVVSM](#)
- You will then see that Unit 3 is assigned to you. Begin working on the Unit 3 assignments posted below.
- There are a few written assignments that you will submit on Showbie.  
The Showbie code for the class is: [7VKC2](#)
- You will be coding in Javascript using the drag-and-drop method in AppLab. Feel free to go between the text and box coding method as you feel comfortable.
- Coding assignments must be done on a PC

### Lesson 1: Introduction to Apps

1. Watch the video: <https://www.youtube.com/watch?v=mCq8-xTH7jA&feature=youtu.be>
2. Go through all of the examples and follow the directions posted on Code.org
3. Complete the Matching Assessment: Vocabulary

### Lesson 2: Introduction to Design Mode

1. Follow the instructions as you go through each of the sections to learn AppLab Design Mode
2. Respond to the Naming Element IDs reflection

### Lesson 3: Designing an App Part 1

1. Download the App Development Planning Guide on Showbie
2. Follow the instructions and create your prototype
3. Build your screens in AppLab

### Lesson 4: The Need for Programming Languages

1. Watch Video: <https://youtu.be/xfBWk4nw440>
2. Watch Video: <https://youtu.be/EGQh5SZctaE>
3. Answer question at the end of Lesson

### Lesson 5: Intro to Programming

1. Complete all the steps in AppLab
2. Answer question at the end of Lesson

### Lesson 6: Debugging

1. Complete all the steps in AppLab
2. Answer question at the end of Lesson

### Lesson 7: Designing an App Part 2

1. Start programming your app. *You will NOT be working with other students.*
2. Complete all programming assignments on AppLab on your own
3. For this final project, you will be creating an app *You will NOT be working with other students.*
4. Submit the App Development Planning Guide on Showbie in the Unit 3 Lesson 7 folder along with the link to your app.

You will be tested on Unit 3 at the start of the 1st Quarter.  
Review the videos and the presentation, along with the content on Code.org

Questions? Email: [hfalcon@pacehs.com](mailto:hfalcon@pacehs.com)

You will be given a grade for assignment and tested on the **vocabulary terms attached below** on the *first full day of class*.

## VOCABULARY TERMS

1. **==**: The equality operator (sometimes read: "equal equal") is used to compare two values, and returns a Boolean (true/false). *Avoid confusion with the assignment operator "=",*
2. **Abstraction**: Pulling out specific differences to make one solution work for multiple problems. (watch this video: <https://youtu.be/L1-zCdrx8Lk>)
3. **Aggregation**: a computation in which rows from a data set are grouped together and used to compute a single value of more significant meaning or measurement. Common aggregations include: Average, Count, Sum, Max, Median, etc.
4. **Algorithm**: A precise sequence of instructions for processes that can be executed by a computer
5. **API**: a collection of commands made available to a programmer
6. **Array**: A data structure in JavaScript used to represent a list.
7. **ASCII**: ASCII - American Standard Code for Information Interchange. ASCII is the universally recognized raw text format that any computer can understand.
8. **asymmetric encryption**: used in public key encryption, it is scheme in which the key to encrypt data is different from the key to decrypt.
9. **Bandwidth**: Transmission capacity measure by bit rate
10. **Binary**: A way of representing information using only two options.
11. **Bit rate**: (sometimes written bitrate) the number of bits that are conveyed or processed per unit of time. e.g. 8 bits/sec.
12. **Bit**: A contraction of "Binary Digit". A bit is the single unit of information in a computer, typically represented as a 0 or 1.
13. **Boolean Expression**: in programming, an expression that evaluates to True or False.
14. **Boolean**: A single value of either TRUE or FALSE
15. **Caesar Cipher**: a technique for encryption that shifts the alphabet by some number of characters
16. **Callback function**: a function specified as part of an event listener; it is written by the programmer but called by the system as the result of an event trigger.
17. **Canvas**: a user interface element to use in HTML/JavaScript which acts as a digital canvas, allowing the programmatic drawing and manipulation of pixels, basic shapes, figures and images.
18. **Cipher**: the generic term for a technique (or algorithm) that performs encryption
19. **code**: (v) to write code, or to write instructions for a computer.
20. **Computationally Hard**: a "hard" problem for a computer is one in which it cannot arrive at a solution in a reasonable amount of time.
21. **Concatenate**: to link together or join. Typically used when joining together text Strings in programming (e.g. "Hello, "+name)
22. **Conditionals**: Statements that only run under certain conditions.
23. **Cracking encryption**: When you attempt to decode a secret message without knowing all the specifics of the cipher, you are trying to "crack" the encryption.
24. **Data Type**: All values in a programming language have a "type" - such as a Number, Boolean, or String - that dictates how the computer will interpret it. For example, 7+5 is interpreted differently from "7"+"5"
25. **Debugging**: Finding and fixing problems in your algorithm or program.
26. **Decryption**: a process that reverses encryption, taking a secret message and reproducing the original plain text
27. **DNS**: The service that translates URLs to IP addresses.
28. **Documentation**: a description of the behavior of a command, function, library, API, etc.
29. **Encryption**: a process of encoding messages to keep them secret, so only "authorized" parties can read it.

30. **Event handling:** an overarching term for the coding tasks involved in making a program respond to events by triggering functions.
31. **Event listener:** a command that can be set up to trigger a function when a particular type of event occurs on a particular UI element.
32. **Event-driven program:** a program designed to run blocks of code or functions in response to specified events (e.g. a mouse click)
33. **Event:** An action that causes something to happen.
34. **Expression:** Any valid unit of code that resolves to a value.
35. **For Loop:** A particular kind of looping construct provided in many languages. Typically, a for loop defines a counting variable that is checked and incremented on each iteration in order to loop a specific number of times.
36. **for loop:** A typical looping construct designed to make it easy to repeat a section of code using a counter variable. The for loop combines the creation of a variable, a boolean looping condition, and an update to the variable in one statement.
37. **Function:** A piece of code that you can easily call over and over again.
38. **Global Variable:** A variable whose scope is "global" to the program, it can be used and updated by any part of the code. Its global scope is typically derived from the variable being declared (created) outside of any function, object, or method.
39. **Heuristic:** a problem-solving approach (algorithm) to find a satisfactory solution where finding an optimal or exact solution is impractical or impossible.
40. **HTTP:** HyperText Transfer Protocol - the protocol used for transmitting web pages over the Internet
41. **IETF:** Internet Engineering Task Force - develops and promotes voluntary Internet standards and protocols, in particular the standards that comprise the Internet protocol suite (TCP/IP).
42. **If-Statement:** The common programming structure that implements "conditional statements".
43. **Image:** A type of data used for graphics or pictures.
44. **Innovation:** A novel or improved idea, device, product, etc, or the development thereof.
45. **Internet:** A group of computers and servers that are connected to each other.
46. **IP Address:** A number assigned to any item that is connected to the Internet.
47. **Iterate:** To repeat in order to achieve, or get closer to, a desired goal.
48. **Key Event:** in JavaScript, an event triggered by pressing or releasing a key on the keyboard. For example: "keyup" and "keydown" are event types you can specify. Use event.key - from the "event" parameter of the onEvent callback function - to figure out which key was pressed.
49. **Latency:** Time it takes for a bit to travel from its sender to its receiver.
50. **Library:** a collection of commands / functions, typically with a shared purpose
51. **List:** A generic term for a programming data structure that holds multiple items.
52. **Local Variable:** A variable with local scope is one that can only be seen, used and updated by code within the same scope. Typically, this means the variable was declared (created) inside a function -- includes function parameter variables.
53. **Loop:** The action of doing something over and over again.
54. **Lossless Compression:** a data compression algorithm that allows the original data to be perfectly reconstructed from the compressed data.
55. **Lossy Compression:** (or irreversible compression) a data compression method that uses inexact approximations, discarding some data to represent the content. Most commonly seen in image formats like .jpg.
56. **metadata:** is data that describes other data. For example, a digital image may include metadata that describe the size of the image, number of colors, or resolution.
57. **Models and Simulations:** a program which replicates or mimics key features of a real-world event in order to investigate its behavior without the cost, time, or danger of running an experiment in real life.
58. **modulo:** a mathematical operation that returns the remainder after integer division. Example:  $7 \text{ MOD } 4 = 3$
59. **Moore's Law:** a predication made by Gordon Moore in 1965 that computing power will double every 1.5-2 years, it has remained more or less true ever since.

60. **Net Neutrality:** the principle that all Internet traffic should be treated equally by Internet Service Providers.
61. **One-pager:** A business/corporate term for a one-page document that summarizes a large issue, topic or plan.
62. **Packets:** Small chunks of information that have been carefully formed from larger chunks of information.
63. **Parameter:** An extra piece of information that you pass to the function to customize it for a specific need.
64. **Pivot Table:** in most spreadsheet software, it is the name of the tool used to create summary tables.
65. **pixel:** short for "picture element" it is the fundamental unit of a digital image, typically a tiny square or dot which contains a single point of color of a larger image.
66. **Private Key:** In an asymmetric encryption scheme the decryption key is kept private and never shared, so only the intended recipient has the ability to decrypt a message that has been encrypted with a public key.
67. **Protocol:** A set of rules governing the exchange or transmission of data between devices.
68. **Public Key Encryption:** Used prevalently on the web, it allows for secure messages to be sent between parties without having to agree on, or share, a secret key. It uses an asymmetric encryption scheme in which the encryption key is made public, but the decryption key is kept private.
69. **Random Substitution Cipher:** an encryption technique that maps each letter of the alphabet to a randomly chosen other letters of the alphabet.
70. **Return Value:** A value sent back by a function to the place in the code where the function was called from - typically asking for value (e.g. `getText(id)`) or the result of a calculation or computation of some kind. Most programming languages have many built-in functions that return values, but you can also write your own.
71. **RGB:** the RGB color model uses varying intensities of (R)ed, (G)reen, and (B)lue light are added together in to reproduce a broad array of colors.
72. **Selection:** A generic term for a type of programming statement (usually an if-statement) that uses a Boolean condition to determine, or select, whether or not to run a certain block of statements.
73. **String:** Any sequence of characters between quotation marks (ex: "hello", "42", "this is a string!").
74. **Summary Table:** a table that shows the results of aggregations performed on data from a larger data set, hence a "summary" of larger data. Spreadsheet software typically calls them "pivot tables".
75. **TCP:** Transmission Control Protocol - provides reliable, ordered, and error-checked delivery of a stream of packets on the internet. TCP is tightly linked with IP and usually seen as TCP/IP in writing.
76. **UI Elements:** on-screen objects, like buttons, images, text boxes, pull down menus, screens and so on.
77. **URL:** An easy-to-remember address for calling a web page (like `www.code.org`).
78. **User Interface:** The visual elements of a program through which a user controls or communications the application. Often abbreviated UI.
79. **Variable Scope:** dictates what portions of the code can "see" or use a variable, typically derived from where the variable was first created. (See Global v. Local)
80. **Variable:** A placeholder for a piece of information that can change.
81. **while loop:** a programming construct used to repeat a set of commands (loop) as long as (while) a boolean condition is true.