California Standards Alignment with CodeX Python

	Unit 1	Unit 2	Unit 3
Computing Systems			
6-8.CS.1 Design modifications to computing devices in order to improve the ways users interact with the devices.			
6-8.CS.2 Design a project that combines hardware and software components to collect and exchange data.			
6-8.CS.3 Systematically apply troubleshooting strategies to identify and resolve hardware and software problems in computing systems.	[1]		
Networks and the Internet			
6-8.NI.4 Model the role of protocols in transmitting data across networks and the Internet.			
6-8.NI.5 Explain potential security threats and security measures to mitigate threats.			
6-8.NI.6 Apply multiple methods of information protection to model the secure transmission of information.			
Data and Analysis			
6-8.DA.7 Represent data in multiple ways.			
6-8.DA.8 Collect data using computational tools and transform the data to make it more useful.			
6-8.DA.9 Test and analyze the effects of changing variables while using computational models.		[2]	
Algorithms and Programming			
6-8.AP.10 Use flowcharts and/or pseudocode to design and illustrate algorithms that solve complex problems.	[3]		
6-8.AP.11 Create clearly named variables that store data, and perform operations on their contents.	[4]		
6-8.AP.12 Design and iteratively develop programs that combine control structures and use compound conditions.		[5]	
6-8.AP.13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.	[6]		
6-8.AP.14 Create procedures with parameters to organize code and make it easier to reuse.	[7]		
6-8.AP.15 Seek and incorporate feedback from team members and users to refine a solution that meets user needs.			
6-8.AP.16 Incorporate existing code, media, and libraries into original programs, and give attribution.	[8]		
6-8.AP.17 Systematically test and refine programs using a range of test cases.			
6-8.AP.18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.			
6-8.AP.19 Document programs in order to make them easier to use, read, test, and debug.	[9]		
Impacts of Computing			
6-8.IC.20 Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.			

California Standards Alignment with CodeX Python

	Unit 1	Unit 2	Unit 3
6-8.IC.21 Discuss issues of bias and accessibility in the design of existing technologies.			
6-8.IC.22 Collaborate with many contributors when creating a computational artifact.			
6-8.IC.23 Compare tradeoffs associated with licenses for computational artifacts to balance the protection of the creators' rights and the ability for others to use and modify the artifacts.			
6-8.IC.24 Compare tradeoffs between allowing information to be public and keeping information private and secure.			

California Standards Alignment with CodeX Python

	Unit 1	Unit 2	Unit 3
Computing Systems			
9-12.CS.1 Describe ways in which abstractions hide the underlying implementation details of computing systems to simplify user experiences.			
9-12.CS.2 Compare levels of abstraction and interactions between application software, system software, and hardware.			
9-12.CS.3 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.	[10]		
Networks and the Internet			
9-12.NI.4 Describe issues that impact network functionality.			
9-12.NI.5 Describe the design characteristics of the Internet.			
9-12.NI.6 Compare and contrast security measures to address various security threats.			
9-12.NI.7 Compare and contrast cryptographic techniques to model the secure transmission of information.			
Data and Analysis			
9-12.DA.8 Translate between different representations of data abstractions of real-world phenomena, such as characters, numbers, and images.			
9-12.DA.9 Describe tradeoffs associated with how data elements are organized and stored.			
9-12.DA.10 Create data visualizations to help others better understand real-world phenomena.			
9-12.DA.11 Refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process.			
Algorithms and Programming			
9-12.AP.12 Design algorithms to solve computational problems using a combination of original and existing algorithms.	[11]		
9-12.AP.13 Create more generalized computational solutions using collections instead of repeatedly using simple variables.		[12]	
9-12.AP.14 Justify the selection of specific control structures by identifying tradeoffs associated with implementation, readability, and performance.			
9-12.AP.15 Iteratively design and develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.	[13]		
9-12.AP.16 Decompose problems into smaller subproblems through systematic analysis, using constructs such as procedures, modules, and/or classes.	[14]		
9-12.AP.17 Create computational artifacts using modular design.		[15]	
9-12.AP.18 Systematically design programs for broad audiences by incorporating feedback from users.			
9-12.AP.19 Explain the limitations of licenses that restrict use of computational artifacts when using resources such as libraries.			

California Standards Alignment with CodeX Python

	Unit 1	Unit 2	Unit 3
9-12.AP.20 Iteratively evaluate and refine a computational artifact to enhance its performance, reliability, usability, and accessibility.		[16]	
9-12.AP.21 Design and develop computational artifacts working in team roles using collaborative tools.			
9-12.AP.22 Document decisions made during the design process using text, graphics, presentations, and/or demonstrations in the development of complex programs.	[17]		
Impacts of Computing			
9-12.IC.23 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.			
9-12.IC.24 Identify impacts of bias and equity deficit on design and implementation of computational artifacts and apply appropriate processes for evaluating issues of bias.			
9-12.IC.25 Demonstrate ways a given algorithm applies to problems across disciplines.			
9-12.IC.26 Study, discuss, and think critically about the potential impacts and implications of emerging technologies on larger social, economic, and political structures, with evidence from credible sources.			
9-12.IC.27 Use collaboration tools and methods to increase connectivity with people of different cultures and careers.			
9-12.IC.28 Explain the beneficial and harmful effects that intellectual property laws can have on innovation.			
9-12.IC.29 Explain the privacy concerns related to the collection and generation of data through automated processes.			
9-12.IC.30 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics.			

[1] Mission 2 introduces troubleshooting techniques. There are more in the teachers manual

[2] 7.3 shows in the debugger variable changes as the program runs

[3] Teachers manual begins suggesting this with the remixes in Mission 4

[4] 3.8 begins the use of variables
5.5 discusses variable descriptive names

[5] Mission 9 begins the use of compound conditionals

[6] Code Tracing Charts do this as suggested in the teachers manual

[7] maintenance and readability are discussed in 5.5
Code Tracing charts can begin helping create notes for future use

[8] All programs in our lessons use libraries and every time a new one is introduced, it is explained.

[9] maintenance and readability are discussed in 5.5

[10] Code Tracing charts are designed to help refer back to when errors you have seen before occur again. You refer to your chart to know how you fixed it before.

[11] Mission 4 begins the use of algorithms. It does not identify itself as an algorithm though so the teacher would need to insert that lesson.

[12] Mission 7 begins the use of lists

[13] Remixes do this depending on the rubric the teacher creates for the remix.

[14] Flowcharts and pseudocodes suggested in the teachers manual covers this

[15] Mission 7 flowcharts would have to use modular design to break down the branching and loops that are coded.

[16] This is done in the remixes

[17] flowcharts, pseudocodes, and code tracing charts cover this