Tennessee CS Standards Alignment with Python with Robots Curriculum				
	Unit 1	Unit 2	Unit 3	Unit 4
Coding and Computer Programming				
CCP.1 Identify the advantages, disadvantages and unintended consequences of computing devices.				
CCP.2 Analyze the relationship between human and computer interactions to improve the device. For example, student A watches student B use a simple communication device. Student A updates the tool for improved use.				
CCP.3 Identify and describe multiple considerations and tradeoffs when designing or selecting computing system, such as functionality, cost, size, speed, accessibility, and aesthetics.				
CCP.4 Construct optimized models of computing systems.				
CCP.5 Create structured processes to troubleshoot problems with computing systems.				
CCP.6 Define protocols in relation to a set of rules.				
CCP.7 Construct protocols that can be used to share information between people or devices. For example, a binary communication protocol using lights.				
CCP.8 Compare the relative strengths and weaknesses of unique protocols considering security, speed, and reliability.				
CCP.9 Create models of networks that include packets and domain name server (DNS).				
CCP.10 Identify steps to ensure security measures are in place to safeguard online information				
CCP.11 Create cyphers to encrypt data that can be transferred between users.				
CCP.12 Explain how encryption can be used to safeguard data that is sent across a network.				
CCP.13 Evaluate the accuracy and precision of various forms of data collection.				
CCP.14 Identify and define the limiting factors to specific forms of data collection.				
CCP.15 Describe how different formats of stored data represent tradeoffs between quality and size.				
CCP.16 Represent data using different encoding schemes, such as binary, Unicode, Morse code, shorthand, student-created codes.				
CCP.17 Explain the processes used to collect, transform, and analyze data to solve a problem using computational tools.				
CCP.18 Revise variables and constants in computational models to more accurately reflect real-world systems. For example in an ecosystem model, introducing predators as a new variable.				
CCP.19 Solicit and integrate peer feedback as appropriate to develop or refine a program.				
CCP.20 Compare different algorithms that may be used to solve the same problem in terms of their speed, clarity, and size.				
CCP.21 Provide proper attribution when code is borrowed or built upon.				
CCP.22 Interpret the flow of execution of algorithms and predict their outcomes.				
CCP.23 Design, develop, and present computational artifacts such as mobile applications that address social problems both independently and collaboratively.				
CCP.24 Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches. (Clarification: At this level, students may use block- based and/or text-based programming languages.)				
CCP.25 Identify the purpose of variables in relation to programming.				
CCP.26 Create variables that represent different types of data and manipulate their values.				
CCP.27 Define and use procedures that hide the complexity of a task and can be reused to solve similar tasks. (Clarification: Students use and modify, but do not necessarily create, procedures with parameters.)				
CCP.28 Decompose a problem into parts and create solutions for each part.				
CCP.29 Use an iterative design process (e.g., define the problem, generate ideas, build, test, and improve solutions) to solve problems, both independently and collaboratively.				
CCP.30 Analyze the positive and negative impacts of computing technology.				
CCP.31 Recognize there are tradeoffs in computing.				
CCP.32 Explain how social interactions can allow for multiple viewpoints.				
CCP.33 Demonstrate an understanding of digital security.				