7th grade computer science

	Essential Questions	Content/Materials	Skills	Assessments
Hex 1	• What strategies and processes can I use to become a more effective problem solver? • How do computers help people to solve problems? • How do people and computers approach problems differently? • What does a computer need from people in order to solve problems effectively?	Code.org curriculum	AP - Algorithms & Programming • 1B-AP-08 - Compare and refine multiple algorithms for the same task and determine which is the most appropriate. • 1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. • 1B-AP-16 - Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation and review stages of program development. • 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. • 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs. • 2-AP-17 - Systematically test and refine programs using a range of test cases. • 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. CS - Computing Systems • 1B-CS-01 - Describe how internal and external parts of computing devices function to form a system. • 1B-CS-02 - Model how computer hardware and software work together as a system to accomplish tasks. • 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. IC - Impacts of Computing • 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's	Periodic submissions of current work, formative assessment throughout lectures and work time

			everyday activities and career options.	
Hex 2	• What strategies and processes can I use to become a more effective problem solver?	Code.org curriculum	AP - Algorithms & Programming • 1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. • 1B-AP-15 - Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.. • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs IC - Impacts of Computing • 1B-IC-18 - Discuss computing technologies that have changed the world and express how those technologies influence, and are influenced by, cultural practices. • 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. • 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure. NI - Networks & the Internet • 1B-NI-05 - Discuss real-world cybersecurity problems and how personal information can be protected.	Periodic submissions of current work, formative assessment throughout lectures and work time
Hex 3	• Why do people create websites? • How can text	Code.org curriculum	AP - Algorithms & Programming • 1B-AP-12 - Modify, remix or incorporate portions of an existing program into one's own work, to develop something new or add more advanced	Periodic submissions of current work, formative assessment throughout

	communicate content and structure on a web page? ● How can I incorporate content I find online into my own webpage? ● What strategies can I use when coding to find and fix issues?		features. ● 1B-AP-15 - Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended. ● 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. ● 2-AP-17 - Systematically test and refine programs using a range of test cases. ● 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. ● 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. ● 3A-AP-20 - Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries. IC - Impacts of Computing ● 1B-IC-21 - Use public domain or creative commons media and refrain from copying or using material created by others without permission. ● 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. ● 2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies. ● 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure.	lectures and work time
Hex 4	● What is a computer program? ● What are the core features of most programming languages? ● How does programming enable creativity and individual expression? ● What practices and strategies will help me	Code.org curriculum	AP - Algorithms & Programming ● 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. ● 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. ● 2-AP-12 - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. ● 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.	Periodic submissions of current work, formative assessment throughout lectures and work time

	as I write programs?		• 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. • 2-AP-17 - Systematically test and refine programs using a range of test cases. • 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. IC - Impacts of Computing • 2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies.	
Hex 5	• How do software developers manage complexity and scale? • How can programs be organized so that common problems only need to be solved once? • How can I build on previous solutions to create even more complex behavior?	Code.org curriculum	**AP - Algorithms & Programming** • **2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms.** • **2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values.** • **2-AP-12 - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.** • **2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.** • **2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse.** • **2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.** • **2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution.** • **2-AP-17 - Systematically test and refine programs using a range of test cases.** • **2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.** • **2-AP-19 - Document programs in order to make them easier to follow, test, and debug.**	Periodic submissions of current work, formative assessment throughout lectures and work time

Hex 6	• How do designers identify the needs of their user? • How can we ensure that a user's needs are met by our designs? • What processes will best allow us to efficiently create, test, and iterate upon our designs?	Code.org curriculum	AP - Algorithms & Programming • 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. • 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. • 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse. • 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs. • 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. • 2-AP-17 - Systematically test and refine programs using a range of test cases. • 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. • 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. CS - Computing Systems • 2-CS-01 - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices. • 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. DA - Data & Analysis • 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. • 2-DA-09 - Refine computational models based on the data they have generated. IC - Impacts of Computing • 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. • 2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies.	Periodic submissions of current work, formative assessment throughout lectures and work time

			• 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact.	