TOWNSHIP OF UNION PUBLIC SCHOOLS

Grade 6
Computer Literacy

Adopted February 15, 2022

Mission Statement

The mission of the Township of Union Public Schools is to build on the foundations of honesty, excellence, integrity, strong family, and community partnerships. We promote a supportive learning environment where every student is challenged, inspired, empowered, and respected as diverse learners. Through cultivation of students' intellectual curiosity, skills and knowledge, our students can achieve academically and socially, and contribute as responsible and productive citizens of our global community.

Philosophy Statement

The Township of Union Public School District, as a societal agency, reflects democratic ideals and concepts through its educational practices. It is the belief of the Board of Education that a primary function of the Township of Union Public School System is to formulate a learning climate conducive to the needs of all students in general, providing therein for individual differences. The school operates as a partner with the home and community.

Course Description

All students receive computer science and design thinking instruction from Kindergarten through grade 12. The study of these disciplines focuses on a deep understanding of concepts that enable students to think critically and systematically about leveraging technology to solve local and global issues. Authentic learning experiences that enable students to apply content knowledge, integrate concepts across disciplines, develop computational thinking skills, acquire and incorporate varied perspectives, and communicate with diverse audiences about the use and effects of computing prepares New Jersey students for college and careers.

The Township of Union Public Schools Grade 6 Computer Science Curriculum provides a highly interactive and collaborative introduction to the field of computer science as framed within the broader pursuit of utilizing critical thinking skills to solve complex problems. The following units explore these processes while providing students with the opportunity to create programmatic images, animations, interactive art, and games.

Course Description

All students receive computer science and design thinking instruction from Kindergarten through grade 12. The study of these disciplines focuses on a deep understanding of concepts that enable students to think critically and systematically about leveraging technology to solve local and global issues. Authentic learning experiences that enable students to apply content knowledge, integrate concepts across disciplines, develop computational thinking skills, acquire and incorporate varied perspectives, and communicate with diverse audiences about the use and effects of computing prepares New Jersey students for college and careers.

The Township of Union Public Schools Grade 6 Computer Science Curriculum provides a highly interactive and collaborative introduction to the field of computer science as framed within the broader pursuit of utilizing critical thinking skills to solve complex problems. The following units explore these processes while providing students with the opportunity to create programmatic images, animations, interactive art, and games.

Curriculum Units/Pacing Guide

Unit # / Title	Number of Days
Unit 1: Problem Solving and Computing	05 days
Unit 2: Interactive Animations and Games	25 days

Unit Standards Overview

Overview	Standards	Unit Skills Focus	Content-Specific Practices (when applicable)

Unit 1
Problem Solving and Computing
The Problem Solving and Computing unit is a highly interactive and collaborative introduction to the field of computer science, as framed within the broader pursuit of solving problems. Through a series of puzzles, challenges, and real world scenarios, students are introduced to a problem solving process that they will return to repeatedly throughout the course. Students then learn how computers input, output, store, and process information to help humans solve problems within the context of apps. The unit concludes with students designing an app that helps solve a problem of their choosing.
	NJSLS – Computer Science and Design Thinking (See below)	By the end of the unit, students should be able to identify the defined characteristics of a computer and how it is used to solve information problems. They should be able to use a structured problem solving process to address problems and design solutions that use computing technology. The unit also serves to build a collaborative classroom environment where students view computer science as relevant, fun, and empowering.	• Input vs Output
			• Processing
			• Storage
			• Bug
			• Debugging
			• Javascript
			• Shapes
			  ○ Stroke
			  ○ Fill
			• Parameters
			• Variables
			• Random Function
			• Sprites
			• Sprite Properties
			• Sprite Movement
			• Sequence and Layering
			• Loops (Function Draw)
			• Counter Pattern
			• Conditionals
			• Keyboard and Mouse Input
			• Velocity
			• Collision Detection
			• Collision
			• 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 as I write programs?

Suggested Resources *Provide links to specific resources/activities*	**Code.org Unit 1 Lesson Guide:** https://studio.code.org/s/csd1-2021?section_id=3522356 **Code.org Unit 1 Resources:** https://studio.code.org/s/csd1-2021/resources **Code.org Unit 1 Teacher Forum:** https://forum.code.org/tags/c/csd/236/csd-unit-1		
Unit 2: Interactive Animations and Games In the Interactive Animations and Gamesunit, students build on their coding experience as they create programmatic images, animations, interactive art, and games. Starting off with simple, primitive shapes and	NJSLS – Computer Science and Design Thinking (See below)	By the end of this unit, students should be able to create an interactive animation or game that includes basic programming concepts such as control structures, variables, user input, and randomness. they should manage this task by working with others to	

building up to more sophisticated sprite-based games, students become familiar with the programming concepts and the design process computer scientists use daily. They then learn how these simpler constructs can be combined to create more complex programs. In the final project, students develop a personalized, interactive program. Along the way, they practice design, testing, and iteration, as they come to see that failure and debugging are an expected and valuable part of the programming process.		break it down using objects (sprites) and functions. Throughout the process, they should give and respond constructively to peer feedback, and work with their teammates to complete a project. Students should leave this unit viewing themselves as computer programmers and see programming as a fun and creative form of expression.	• 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?
Suggested Resources *Provide links to specific resources/activities*	Code.org Unit 3 Lesson Guide: https://studio.code.org/s/csd3-2021?section_id=3522356 CSD: Guide to Debugging: https://docs.google.com/document/d/1-mXz53CAQ1dy3-RVGkoUnR0hfAwdWeza4kch6NAvLb8/preview CSD: Guide to Differentiation: https://docs.google.com/document/d/15Y6Mo3fMUtHq16SSskABgcYE-5xfk-Ewv0z_o4uaLdI/preview Code.org Game Lab: https://studio.code.org/projects/gamelab/jQ7HayyW_NAX-xdyURI-EJtKsIM2Vuy4hyeWOzrHmk8/edit Code.org Unit 3 Teacher Forum: https://forum.code.org/tags/c/csd/236/csd-unit-3		

Curricular Units

Unit 1: Problem Solving and Computing			
Content Standards	**Critical Knowledge & Skills** ("Unpacked" Standards)	**Content-Specific Practices** (when applicable)	**Standard Mastery Examples** *When possible, provide links to specific samples/ documents/ assignments/etc.*
8.1.5.AP.1	Different algorithms can achieve the same result. Some algorithms are more appropriate for a specific use than others.	• Compare and refine multiple algorithms for the same task and determine which is the most appropriate.	• Lesson 1: Intro to Problem Solving • Lesson 2: The Problem Solving Process • Lesson 3: Exploring Problem Solving • Lesson 9: Intro to Problem Solving - Newspaper Table (Alternate Lesson 1) • Lesson 10: Intro to Problem Solving - Spaghetti Bridge (Alternate Lesson 1) • Lesson 11: Intro to Problem Solving - Paper Tower (Alternate Lesson 1) • Lesson 12: Exploring Problem Solving - Animals Theme (Alternate Lesson 3) • Lesson 13: Exploring Problem Solving - Games Theme (Alternate Lesson 3)
8.1.5.AP.4	Programs can be broken down into smaller parts to facilitate their design, implementation, and review. Programs can also be created by incorporating smaller portions of programs that already exist.	• Break down problems into smaller, manageable sub-problems to facilitate program development.	• Lesson 1: Intro to Problem Solving • Lesson 2: The Problem Solving Process • Lesson 3: Exploring Problem Solving • Lesson 9: Intro to Problem Solving - Newspaper Table (Alternate Lesson 1) • Lesson 10: Intro to Problem Solving - Spaghetti Bridge (Alternate Lesson 1)

			• Lesson 11: Intro to Problem Solving - Paper Tower (Alternate Lesson 1) • Lesson 12: Exploring Problem Solving - Animals Theme (Alternate Lesson 3) • Lesson 13: Exploring Problem Solving - Games Theme (Alternate Lesson 3)
8.2.5.ED.2	Engineering design is a systematic and creative process of communicating and collaborating to meet a design challenge.	• Collaborate with peers to collect information, brainstorm to solve a problem, and evaluate all possible solutions to provide the best results with supporting sketches or models.	• Lesson 1: Intro to Problem Solving • Lesson 3: Exploring Problem Solving • Lesson 9: Intro to Problem Solving - Newspaper Table (Alternate Lesson 1) • Lesson 10: Intro to Problem Solving - Spaghetti Bridge (Alternate Lesson 1) • Lesson 11: Intro to Problem Solving - Paper Tower (Alternate Lesson 1) • Lesson 12: Exploring Problem Solving - Animals Theme (Alternate Lesson 3) • Lesson 13: Exploring Problem Solving - Games Theme (Alternate Lesson 3)
8.1.5.CS.1	Computing devices may be connected to other devices to form a system as a way to extend their capabilities.	• Model how computing devices connect to other components to form a system.	• Lesson 4: What is a Computer? • Lesson 5: Input and Output • Lesson 6: Processing
8.1.5.CS.2	Software and hardware work together as a system to accomplish tasks (e.g., sending, receiving, processing, and storing units of information).	• Model how computer software and hardware work together as a system to accomplish tasks.	• Lesson 5: Input and Output • Lesson 6: Processing
8.1.2.AP.1	Individuals develop and follow directions as part of daily life.	• Model daily processes by creating and following algorithms to complete tasks.	• Lesson 6: Processing • Lesson 7: Storage

			• Lesson 8: Project - Propose an App
8.1.5.AP.6	Individuals develop programs using an iterative process involving design, implementation, testing, and review.	• Develop programs using an iterative process, implement the program design, and test the program to ensure it works as intended.	• Lesson 6: Processing
8.2.5.NT.2	Technology innovation and improvement may be influenced by a variety of factors. Engineers create and modify technologies to meet people's needs and wants; scientists ask questions about the natural world.	• Identify new technologies resulting from the demands, values, and interests of individuals, businesses, industries, and societies.	• Lesson 7: Storage
8.2.5.ED.2:	Engineering design is a systematic and creative process of communicating and collaborating to meet a design challenge.	• Collaborate with peers to collect information, brainstorm to solve a problem, and evaluate all possible solutions to provide the best results with supporting sketches or models.	• Lesson 8: Project - Propose an App
8.2.8.ED.7	Engineering design requirements and specifications involve making trade-offs between competing requirements and desired design features.	• Design a product to address a real-world problem and document the iterative design process, including decisions made as a result of specific constraints and trade-offs (e.g., annotated sketches).	• Lesson 8: Project - Propose an App

Unit 1 Assessment Plan	
Formative Assessment *When possible, provide links to specific samples/ documents/ assignments/etc.*	**Summative Assessment** *When possible, provide links to specific samples/ documents/ assignments/etc.*
• Observation • Completion of Activity Guides included in CSD Unit 1 Resources	• CSD Unit 1 Post-Project Test • Completion of Activity Guides included in CSD Unit 1 Resources • Lesson 8: Propose an App

Unit 1 Suggested Modifications/Accommodations/Extension Activities		
English Language Learners (ELL) *When possible, provide links to specific samples/ documents/ assignments/etc.*	**Special Education / 504** *When possible, provide links to specific samples/ documents/ assignments/etc.*	**Gifted and Talented** *When possible, provide links to specific samples/ documents/ assignments/etc.*
CS Discoveries Approach to Differentiation "Learning to use resources is a key goal of the course, and given resources provide an opportunity for students to self-differentiate in how they interact with key course content. This may include proactive differentiation, such as printing out resources ahead of time for students. It can also include just-in-time differentiation, such as monitoring students as they reach the end of a project and referring them to additional resource" **Examples of Strategies and Practices that Support English Language Learners:** • Pre-teaching of vocabulary and concepts • Visual learning, including graphic organizers • Text to Speech • Think-pair-share • Cooperative learning groups • Teacher modeling • Pairing students with beginning English language skills with students who have more advanced English language skills • Documentation Resource on code.org	CS Discoveries Approach to Differentiation "In order to meet the needs of a wide variety of learners, CS Discoveries is designed with flexibility that allows teachers to differentiate their instruction at the class and student level." *Refer to students' IEP for specific modifications and accommodations **Examples of Strategies and Practices that Support Students with Disabilities:** • Use of visual and multisensory formats • Use of assisted technology • Use of prompts • Modification of content and student products • Testing accommodations • Authentic assessments	CS Discoveries Approach to Differentiation "Challenge levels are found after the assessment levels in many of the programming lessons. These levels include new code and challenges that go beyond the learning objectives of the lesson. Most also include a "Free Play" option that allows students to use the new skills they have learned in whatever way they choose."

Unit 1 Connections	
NJSLS - Technology *When possible, provide links to specific samples/ documents/ assignments/etc.* Refer to the NJ Technology Standards	**Career Readiness Practices** *When possible, provide links to specific samples/ documents/ assignments/etc.* Refer to the NJ Career Readiness Practices
Technology Standards: Technology standards are embedded throughout all curricular units: Standard 8.1 Computer Science	**Career Ready Practices and Standard 9.1, 9.2, and 9.3 Career Ready Practices:** • CRP2. Apply appropriate academic and technical skills.

• Computer Science, previously a strand entitled 'Computational Thinking: Programming' in standard 8.2 of the 2014 NJSLS-Technology, outlines a comprehensive set of concepts and skills, such as data and analysis, algorithms and programming, and computing systems. Standard 8.2 Design Thinking • This standard, previously standard 8.2 Technology Education of the 2014 NJSLS – Technology, outlines the technological design concepts and skills essential for technological and engineering literacy. The new framework design, detailed previously, includes Engineering Design, Ethics and Culture, and the Effects of Technology on the Natural world among the disciplinary concepts.	• CRP4. Communicate clearly and effectively and with reason. • CRP5. Consider the environmental, social and economic impacts of decisions. • CRP6. Demonstrate creativity and innovation. • CRP7. Employ valid and reliable research strategies. • CRP8. Utilize critical thinking to make sense of problems and persevere in solving them. • CRP9. Model integrity, ethical leadership and effective management. • CRP10. Plan education and career paths aligned to personal goals. • CRP11. Use technology to enhance productivity. • CRP12. Work productively in teams while using cultural global competence.
21st Century Skills *When possible, provide links to specific samples/ documents/ assignments/etc.* Refer to the 21st Century Life and Skills	**Interdisciplinary Connections** *When possible, provide links to specific ELA/Math/Sci/SS standards as well as samples/ documents/ assignments/etc.* Refer to the NJ Student Learning Standards
21st Century Themes • Career Awareness • Career Exploration **21st Century Skills** • Creativity and Innovation (E) • Critical Thinking and Problem Solving (T) (A) • Communication (E) • Collaboration (E) (T)	**Interdisciplinary connections are made across grades and content areas to model the integration of knowledge and skills in the real world.**

Unit 2: Interactive Animations and Games			
Content Standards	**Critical Knowledge & Skills** ("Unpacked" Standards)	**Content-Specific Practices** (when applicable)	**Standard Mastery Examples** *When possible, provide links to specific samples/ documents/ assignments/etc.*
8.1.8.AP.1	Individuals design algorithms that are reusable in many situations. Algorithms that are readable are easier to follow, test, and debug.	Design and illustrate algorithms that solve complex problems using flowcharts and/or pseudocode.	• Lesson 02: Plotting Shapes • Lesson 27: Project - Design a Game
8.1.8.AP.2	Programmers create variables to store data values of different types and perform appropriate operations on their values.	Create clearly named variables that represent different data types and perform operations on their values.	• Lesson 05: Variables • Lesson 06: Random Numbers • Lesson 07: Sprites • Lesson 08: Sprite Properties • Lesson 10: Mini-Project - Captioned Scenes • Lesson 11: The Draw Loop • Lesson 12: Sprite Movement • Lesson 13: Mini-Project - Animation • Lesson 15: Keyboard Input • Lesson 16: Mouse Input • Lesson 17: Project - Interactive Card • Lesson 18: Velocity • Lesson 19: Collision Detection • Lesson 20: Mini-Project - Side Scroller • Lesson 21: Complex Sprite Movement • Lesson 22: Collisions • Lesson 23: Mini-Project - Flyer Game • Lesson 24: Functions • Lesson 25: The Game Design Process • Lesson 26: Using the Game Design Process • Lesson 27: Project - Design a Game

8.1.8.AP.3	Control structures are selected and combined in programs to solve more complex problems.	Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	• Lesson 09: Text • Lesson 11: The Draw Loop • Lesson 12: Sprite Movement • Lesson 14: Conditionals • Lesson 15: Keyboard Input • Lesson 16: Mouse Input • Lesson 18: Velocity • Lesson 19: Collision Detection • Lesson 20: Mini-Project - Side Scroller • Lesson 21: Complex Sprite Movement • Lesson 22: Collisions • Lesson 23: Mini-Project - Flyer Game • Lesson 24: Functions • Lesson 25: The Game Design Process • Lesson 26: Using the Game Design Process • Lesson 27: Project - Design a Game
8.1.8.AP.4	Programs use procedures to organize code and hide implementation details. Procedures can be repurposed in new programs. Defining parameters for procedures can generalize behavior and increase reusability	Decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs.	• Lesson 02: Plotting Shapes • Lesson 03: Drawing in Game Lab • Lesson 04: Shapes and Parameters • Lesson 05: Variables • Lesson 06: Random Numbers • Lesson 07: Sprites • Lesson 08: Sprite Properties • Lesson 09: Text • Lesson 10: Mini-Project - Captioned Scenes • Lesson 11: The Draw Loop • Lesson 12: Sprite Movement • Lesson 13: Mini-Project - Animation • Lesson 14: Conditionals • Lesson 16: Mouse Input • Lesson 17: Project - Interactive Card • Lesson 18: Velocity • Lesson 19: Collision Detection

			• Lesson 20: Mini-Project - Side Scroller • Lesson 21: Complex Sprite Movement • Lesson 22: Collisions • Lesson 23: Mini-Project - Flyer Game • Lesson 24: Functions • Lesson 25: The Game Design Process • Lesson 26: Using the Game Design Process • Lesson 27: Project - Design a Game
8.1.8.AP.5	Programs use procedures to organize code and hide implementation details. Procedures can be repurposed in new programs. Defining parameters for procedures can generalize behavior and increase reusability	Create procedures with parameters to organize code and make it easier to reuse.	• Lesson 24: Functions
8.1.8.AP.6	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community.	Refine a solution that meets users' needs by incorporating feedback from team members and users.	• Lesson 17: Project - Interactive Card • Lesson 27: Project - Design a Game
8.1.8.AP.7	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community.	Design programs, incorporating existing code, media, and libraries, and give attribution	• Lesson 07: Sprites • Lesson 11: The Draw Loop • Lesson 12: Sprite Movement • Lesson 15: Keyboard Input • Lesson 16: Mouse Input • Lesson 17: Project - Interactive Card • Lesson 18: Velocity • Lesson 19: Collision Detection • Lesson 21: Complex Sprite Movement • Lesson 22: Collisions • Lesson 24: Functions • Lesson 25: The Game Design Process • Lesson 26: Using the Game Design Process

			• Lesson 27: Project - Design a Game
8.1.8.AP.6	Individuals develop programs using an iterative process involving design, implementation, testing, and review.	Develop programs using an iterative process, implement the program design, and test the program to ensure it works as intended.	• Lesson 02: Plotting Shapes • Lesson 04: Shapes and Parameters • Lesson 05: Variables • Lesson 06: Random Numbers • Lesson 07: Sprites • Lesson 08: Sprite Properties • Lesson 09: Text • Lesson 10: Mini-Project - Captioned Scenes • Lesson 11: The Draw Loop • • Lesson 12: Sprite Movement • Lesson 13: Mini-Project - Animation • Lesson 14: Conditionals • Lesson 15: Keyboard Input • Lesson 16: Mouse Input • Lesson 17: Project - Interactive Card • Lesson 18: Velocity • Lesson 19: Collision Detection • Lesson 20: Mini-Project - Side Scroller • Lesson 21: Complex Sprite Movement • Lesson 22: Collisions • Lesson 23: Mini-Project - Flyer Game • Lesson 24: Functions • Lesson 25: The Game Design Process • Lesson 26: Using the Game Design Process • Lesson 27: Project - Design a Game
8.1.8.AP.9	Individuals design and test solutions to identify problems taking into consideration the diverse needs of the users and the community.	Document programs in order to make them easier to follow, test, and debug	• Lesson 03: Drawing in Game Lab • Lesson 04: Shapes and Parameters • Lesson 05: Variables

			• Lesson 06: Random Numbers • Lesson 07: Sprites • Lesson 08: Sprite Properties • Lesson 09: Text • Lesson 10: Mini-Project - Captioned Scenes • Lesson 11: The Draw Loop • Lesson 12: Sprite Movement • Lesson 13: Mini-Project - Animation • Lesson 14: Conditionals • Lesson 15: Keyboard Input • Lesson 16: Mouse Input • Lesson 17: Project - Interactive Card • Lesson 18: Velocity • Lesson 21: Complex Sprite Movement • Lesson 22: Collisions • Lesson 23: Mini-Project - Flyer Game • Lesson 24: Functions • Lesson 25: The Game Design Process • Lesson 26: Using the Game Design Process • Lesson 27: Project - Design a Game
8.2.8.ED.7	Engineering design requirements and specifications involve making trade-offs between competing requirements and desired design features.	Design a product to address a real-world problem and document the iterative design process, including decisions made as a result of specific constraints and trade-offs (e.g., annotated sketches).	• Lesson 17: Project - Interactive Card • Lesson 27: Project - Design a Game
8.1.8.IC.2	Advancements in computing technology can change individuals' behaviors. Society is faced with trade-offs due to the increasing globalization and automation that computing brings.	Describe issues of bias and accessibility in the design of existing technologies.	• Lesson 01: Programming for Entertainment

Unit 2 Assessment Plan	
Formative Assessment *When possible, provide links to specific samples/ documents/ assignments/etc.*	**Summative Assessment** *When possible, provide links to specific samples/ documents/ assignments/etc.*
• Observation • Completion of Mini-Project Lessons	• Creation of a game in Lesson 27: Project - Design a Game

Unit 2 Suggested Modifications/Accommodations/Extension Activities		
English Language Learners (ELL) *When possible, provide links to specific samples/ documents/ assignments/etc.*	**Special Education / 504** *When possible, provide links to specific samples/ documents/ assignments/etc.*	**Gifted and Talented** *When possible, provide links to specific samples/ documents/ assignments/etc.*
CS Discoveries Approach to Differentiation "Learning to use resources is a key goal of the course, and given resources provide an opportunity for students to self-differentiate in how they interact with key course content. This may include proactive differentiation, such as printing out resources ahead of time for students. It can also include just-in-time differentiation, such as monitoring students as they reach the end of a project and referring them to additional resource" **Examples of Strategies and Practices that Support English Language Learners:** Pre-teaching of vocabulary and concepts Visual learning, including graphic organizers Text to Speech Think-pair-share Cooperative learning groups Teacher modeling	CS Discoveries Approach to Differentiation "In order to meet the needs of a wide variety of learners, CS Discoveries is designed with flexibility that allows teachers to differentiate their instruction at the class and student level." *Refer to students' IEP for specific modifications and accommodations **Examples of Strategies and Practices that Support Students with Disabilities:** • Use of visual and multisensory formats • Use of assisted technology • Use of prompts • Modification of content and student products • Testing accommodations • Authentic assessments	CS Discoveries Approach to Differentiation "Challenge levels are found after the assessment levels in many of the programming lessons. These levels include new code and challenges that go beyond the learning objectives of the lesson. Most also include a "Free Play" option that allows students to use the new skills they have learned in whatever way they choose."

Pairing students with beginning English language skills with students who have more advanced English language skills Documentation Resource on code.org "Learning to use resources is a key goal of the course, and given resources provide an opportunity for students to self-differentiate in how they interact with key course content. This may include proactive differentiation, such as printing out resources ahead of time for students. It can also include just-in-time differentiation, such as monitoring students as they reach the end of a project and referring them to additional resource" **Examples of Strategies and Practices that Support English Language Learners:** • Pre-teaching of vocabulary and concepts • Visual learning, including graphic organizers • Text to Speech • Think-pair-share • Cooperative learning groups • Teacher modeling • Pairing students with beginning English language skills with students who have more advanced English language skills • Documentation Resource on code.org		

Unit 2 Connections	
NJSLS - Technology *When possible, provide links to specific samples/ documents/ assignments/etc.* Refer to the NJ Technology Standards	**Career Readiness Practices** *When possible, provide links to specific samples/ documents/ assignments/etc.* Refer to the NJ Career Readiness Practices

21st Century Skills *When possible, provide links to specific samples/ documents/ assignments/etc.* Refer to the 21st Century Life and Skills	**Interdisciplinary Connections** *When possible, provide links to specific ELA/Math/Sci/SS standards as well as samples/ documents/ assignments/etc.* Refer to the NJ Student Learning Standards