Career and Technical Education (CTE)

High School/HN Intro to Computer Science

BOARD APPROVAL DATE: 8/17/21

BOARD ADOPTION OF STATE STANDARDS: 9/1/22

Unit Overview (Standards Coverage)				
Unit	**Standards**	**Unit Focus**	**Skills Overview**	**Suggested Pacing**
Unit 1	8.1.12.CS.2&3 8.1.12.AP.1-9 CRP2, CRP6, & CRP11 9.4.12.CT.1&2 ST 2.3&2.5 12.9.3.IT-PRG.4-6	**Computer Basics** **Algorithms and Programming**	**Model interactions and compare the functions of application software, system software, and hardware.** **Students will begin thinking in algorithmic, logical patterns in preparation of developing software (coding).**	**5 Weeks**
Unit 2	CRP2, CRP6, & CRP11 8.1.12.CS.2&3 9.4.12.CT.1&2 8.1.12.AP.1 8.1.12.AP.3 8.1.12.AP.4 ST 2.3&2.5 12.9.3.IT-PRG.4-6	**Programming in Python**	**Developing code in Python that will generate visual output.**	**12 Weeks**
Unit 3	CRP2, CRP6, CRP11 8.1.12.CS.2&3 9.4.12.CT.1&2 8.1.12.AP.1 8.1.12.AP.3 8.1.12.AP.4 ST 2.3&2.5 12.9.3.IT-PRG.4-6	**Object Oriented Programming: Java Syntax, Errors, and Primitive Data**	**Creating working standalone mini programs.**	**5 Weeks**
Unit 4	CRP2, CRP6, CRP11 8.1.12.CS.2&3 9.4.12.CT.1&2 8.1.12.AP.1 8.1.12.AP.3 8.1.12.AP.4 ST 2.3&2.5 12.9.3.IT-PRG.4-6	**Control Statements: Conditionals, If, If-Else, & Loops**	**Gaining control of program execution, allowing programs to make decisions, and varying output based on input.**	**9 Weeks**
Unit 5	CRP2, CRP6, CRP11 8.1.12.CS.2&3 9.4.12.CT.1&2 8.1.12.AP.1 8.1.12.AP.3	**Writing Classes**	**Creating well organized, reusable and shareable code to support the open source nature of programming.**	**8 Weeks**

	8.1.12.AP.4 8.2.12.E.4 8.2.12.E.CS1 ST 2.3&2.5 12.9.3.IT-PRG.4-6			

This document outlines in detail the answers to the following four questions:

1. **What do we want our students to know?**
2. **How do we know if they learned it?**
3. **What do we do if they did not learn it?**
4. **What do we do when they did learn it?**

Unit 1: Computer Basics, Logical Thinking, and Basic Algorithms		
Content & Practice Standards (write in full)	**Interdisciplinary Standards**	**Critical Knowledge & Skills**
CRP2. Apply appropriate academic and technical skills. CRP6. Demonstrate creativity and innovation. 8.1.12.AP.1 - Design algorithms to solve computational problems using a combination of original and existing algorithms. 8.2.12.E.4 - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements). 8.2.12.E.CS1 - Computational thinking and computer programming as tools used in design and engineering. ST 2.3: Apply a currently applicable computer programming language to a process, project, plan or issue as assigned. ST 2.5: Apply a technological, scientific, or mathematical concept (use of algorithms) when communicating with others on issues, plans, processes, problems, or concepts. 12.9.3.IT-PRG.4 - Demonstrate the effective use of software development tools to develop software applications. 12.9.3.IT-PRG.5 - Apply an appropriate software development process to design a software application. 12.9.3.IT-PRG.6 - Program a computer application using the appropriate programming language. 9.4.12.CT.1: Identify problem-solving strategies used in the development of an innovative product or practice 9.4.12.CT.2: Explain the potential benefits of collaborating to enhance critical thinking and problem solving.	**RST.9-10.2**. Determine the central ideas, themes, or conclusions of a text; trace the text's explanation or depiction of a complex process, phenomenon, or concept; provide an accurate summary of the text. **RST.9-10.5.** Analyze the relationships among concepts in a text, including relationships among key terms (e.g., force, friction, reaction force, energy). **RST.11-12.2.** Determine the central ideas, themes, or conclusions of a text; summarize complex concepts, processes, or information presented in a text by paraphrasing them in simpler but still accurate terms. **A -REI** A. Understand solving equations as a process of reasoning and explain the reasoning **F-BF** A. Build a function that models a relationship between two quantities 1. Write a function that describes a relationship between two quantities	Logical Thinking, Problem Solving, and Algorithms ● Understand what an algorithm is as well as develop basic algorithms. ● Enhance problem solving skills through collaboration in solving world problems/logic problems. Computer Systems ● Describe the relationship between hardware and software. ● Define various types of software and how they are used. ● Identify basic computer hardware and explain what it does. ● Explain how computers execute programs and manage data. Understand the role of computer science in the 21st Century including the importance of not only being able to use technology but also to be able to modify and develop technologies.

Unit 1 Computer Basics, Logical Thinking, and Basic Algorithms	
Stage 1 – Desired Results	
Unit Summary	**Core and Supplemental Materials/Resources (open resources)**
Students must understand the machines they are working with. They will develop a basic understanding of hardware and software and how the two need each other. Students will also explore the different things we consider to be computers today and how far we have come from the first computer. Students will understand	Unit Resources Fundamentals of Java AP Computer Science Essentials Java Software Solutions *AP Computer Science CSUnplugged (online)

binary code as well. Finally, they will begin to explore algorithms including what an algorithm is, how to create one, and the role of logical thinking in developing algorithms.	Logic Problems – online, materials shared amongst PLC Internet Resource Links: http://csunplugged.org/activities/ https://www.google.com/culturalinstitute/exhibit/revolution/AR4lLLQ-?position=39%2C0 https://users-mooc.amplify.com/apcs

UNDERSTANDINGS

Students will understand that technology plays a vital role in our society. It is important that students understand the role of technology in society as well as understand not only how to use it, but how to develop and improve it. Students will develop problem solving skills and logical thinking that will help them succeed in any academic or professional field. Students will better understand how technology can be used to improve success in all fields, not just computer science.

Students will know…	**Students will be able to…**
Students will understand what makes something a computer/smart technology. They will learn the relationship between hardware and software, further identifying how one is worthless without the other. They will also know the history/evolution of technology. With an understanding of the technology available to them they will begin to see the importance of creating with the technology not just using it. Students will understand the logical, algorithmic thinking necessary to be successful in Computer Science.	*Students will be able to:* *Logical Thinking, Problem Solving and Algorithms* ● *Understand what an algorithm is as well as develop basic algorithms* ● *Enhance problem solving skills through collaboration in solving world problems/logic problems* *Computer Systems* ● *Describe the relationship between hardware and software.* ● *Define various types of software and how they are used.* ● *Identify basic computer hardware and explain what it does.* ● *Explain how computers execute programs and manage data.* ● *Understand the role of computer science in the 21st Century including the importance of not only being able to use technology but also to be able to modify and develop technologies.*

Stage 2 – Assessment Evidence

Performance Tasks: *Students will begin the year working on problem solving activities that involve collaboration and logical thinking. These activities will demonstrate the importance of collaboration and procedural thinking in computer science. (Activities: Mapping Locations, Crossing a River Logic Problem, Building with Cards, Ordering Cards)* *Algorithm Development Problem: Peanut Butter and Jelly Algorithms – students will develop algorithms to make peanut butter and jelly sandwiches, to be read to a "computer" played by a person doing only what the instructions say (importance of sequential and accurate algorithm development).* **can be modified for allergies (build a paper airplane)* *Students will explore the history of computers by visiting Google Cultural Institute Online which documents the evolution of computers through pictures/descriptions. They will also view the History Channel Video – History of Computers. Next, they will develop visuals of where they think technology is headed.*	Other Evidence **(Alternate Assessments)**: *Vocabulary quizzes* *Chapter Tests* *Completion of assigned worksheets/questions* *Classroom discussion and interaction during activities (collaboration & communication)*

After review of PPT's and classroom discussions students will answer textbook review questions pertaining to hardware and software. They will also complete an activity from CSUnplugged website pertaining to binary code. They will demonstrate the ability to move between Base 10 and Binary.	

Stage 3 – Learning Plan

Students will enhance their problem solving skills and develop the skill set necessary to succeed in CS by solving various logic problems that require logical, process oriented thinking. Their solutions will be discussed by the overall class allowing the final solution to come from the students.

- *Mapping Locations – reinforce the necessity for planning*
- *Crossing a River – forces students to develop the correct algorithm*
- *Building a Structure with Cards – STEM Activity, build something with the materials you have*
- *Ordering Cards – students will have to order cards so cards appear in the correct order based on different scenarios of pulling the cards out of the deck. Backwards design, this is how it should be in the end, can you figure out what the beginning should look like.*

Algorithms: Peanut Butter and Jelly Exercise – students will write out instructions explaining to someone how to make a PB&J Sandwich. Activity will explain what an algorithm is on the most basic level– a process. Instructions will be followed in a demonstration. This activity demonstrates that technology does not make mistakes, programmers/developers do.

Computer Basics – Hardware/Software/Binary
Research of old computers – discussion of evolution (Google Cultural Institute & History of Computers Video)
Develop/Draw Computers of the Future (description of functionality)
Completion of CS Unplugged: Binary Numbers – Count the Dots

- *Demonstration of on/off switch of binary with students holding binary cards.*
- *Students will be given a number and as a class will have to "switch on" the right binary cards to represent that number*

Fundamentals of Java Chapter 1: Summary Questions

Planned Differentiation & Interventions for Tiers I, II, III, ELL, 504s, SPED, and Gift & Talented Students

• Rethink and revise. Dig deeper into ideas at issue (through the faces of understanding). Revise, rehearse, and refine, as needed. Guide students in self-assessment and self-adjustment, based on feedback from inquiry, results, and discussion.

• Evaluate understanding. Reveal what has been understood through final performances and products. Involve students in a final self-assessment to identify remaining questions, set future goals, and point toward new units and lessons.

•Tailor (personalize) the work to ensure maximum interest and achievement. Differentiate the approaches used and provide sufficient options and variety (without compromising goals) to make it most likely that all students will be engaged and effective.

Gifted & Talented:
Students will be assigned more advanced challenges (ex. Sort Cards - instead of one suite students might be required to sort multiple decks). Students will be directed to online websites for more challenging problems and practice. Students will be assigned the lead learner role. (See Tier III)

Tier I:
Students will be assigned more advanced challenges (ex. Sort Cards - instead of one suite students might be required to sort multiple decks). Students will be directed to online websites for more challenging problems and practice.

Tier II:
Students will be provided more guided practice. Instead of free response worksheets students will be given fill in the blank worksheets. Students will be grouped with Tier 1 and Gifted and Talented students when completing group work.

Tier III: Students will be provided more one-on -one instruction. In classes where possible students will also be assigned a lead learner (student that is excelling that can provide assistance whenever necessary). These students will have differentiated test - fill in the bank with word banks. Students will be provided notes and access to all materials.
ELL: Students have access to Google translate. Students may have assignments and assessments printed in their native language if available (many of the unplugged activities have this accommodation). Students will be partnered with other students that speak their native language if possible. Students may take quizzes/exams with an ELL teacher. Additionally, ELL students will be grouped into appropriate tiers and receive those additional accommodations.
504s: Accommodating based on recommendations. Ability for notes and lessons to be unplugged, select grouping, etc. Additionally, 504 students will be grouped into appropriate tiers and receive those additional accommodations.
SPED: Accommodations will be provided according to IEP. Examples: preferential seating, extra time to complete assignments and quizzes, read quiz aloud, copy of notes. Additionally, SPED students will be grouped into appropriate tiers and receive those additional accommodations.

Unit 2 Programming in Python		
Content & Practice Standards	**Suggested Standards for ELA Practice**	**Critical Knowledge & Skills**
CRP2. Apply appropriate academic and technical skills. CRP6. Demonstrate creativity and innovation. CRP11. Use technology to enhance productivity. 8.1.12.CS.2: Model interactions between application software, system software, and hardware. 8.1.12.CS.3: Compare the functions of application software, system software, and hardware. 9.4.12.CT.1: Identify problem-solving strategies used in the development of an innovative product or practice 9.4.12.CT.2: Explain the potential benefits of collaborating to enhance critical thinking and problem solving 8.1.12.AP.1 - Design algorithms to solve computational problems using a combination of original and existing algorithms. 8.1.12.AP.3 - Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice. 8.1.12.AP.4 - Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue. 8.2.12.E.4 - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements). 8.2.12.E.CS1 - Computational thinking and computer programming as tools used in design and engineering. ST 2.3: Apply a currently applicable computer programming language to a process, project, plan or issue as assigned. ST 2.5: Apply a technological, scientific, or mathematical concept (use of algorithms) when communicating with others on issues, plans, processes, problems, or concepts.	**RST.9-10.2.** Determine the central ideas, themes, or conclusions of a text; trace the text's explanation or depiction of a complex process, phenomenon, or concept; provide an accurate summary of the text. **RST.9-10.5.** Analyze the relationships among concepts in a text, including relationships among key terms (e.g., force, friction, reaction force, energy). **RST.11-12.2.** Determine the central ideas, themes, or conclusions of a text; summarize complex concepts, processes, or information presented in a text by paraphrasing them in simpler but still accurate terms. **A -REI** A. Understand solving equations as a process of reasoning and explain the reasoning **F-BF** A. Build a function that models a relationship between two quantities 1. Write a function that describes a relationship between two quantities	*Identify, understand, and apply different programming concepts using Python* ● *Understand the syntax rules of Python* ● *Understand the importance of formatting: spacing, indentations, and commenting out code.* ● *Understand how the computer takes lines of code and puts it in a format that allows it to run a program.* ● *Be able to troubleshoot issues arising due to syntax errors.* ● *Be able to create programs that output visual results* ● *Understand how to use primitive data type variables and arithmetic expressions to make dynamic programs.* ● *Be able to follow/trace the execution of programs that use variables and arithmetic expressions.* ● *Apply concepts to create dynamic stand alone mini programs from start to finish.* ● *Create the necessary variables and algorithms in logical order.* ● *Repeat lines of code efficiently by using loops* ● *Understand the importance of control statements in improving the dynamic nature and efficiency of their code* ● *Create programs that work under various scenarios* ● *Be able to identify what control statements are appropriate at given times* ○ *Decision making statements* ○ *Nested decision* ○ *Loops*

12.9.3.IT-PRG.4 - Demonstrate the effective use of software development tools to develop software applications. 12.9.3.IT-PRG.5 - Apply an appropriate software development process to design a software application. 12.9.3.IT-PRG.6 - Program a computer application using the appropriate programming language.		

Unit 2 Programming in Python

Stage 1 – Desired Results

Unit Summary	Core and Supplemental Materials/Resources (open resources)
Students will be introduced to text based programming by learning the Python programming language. They will understand how text based code can produce visual results. They will come to see that a programming language is no different than a language we speak; it has rules and "punctuation" that must be followed. They will begin to trace and explain completed code as well as type their own stand alone mini programs. Students will be able to write programs and debug basic errors.	Carnegie Mellon CS Academy https://academy.cs.cmu.edu/ Planning Poster Board (mapping programs) Mini-Whiteboards: Tracing Programs Computers - Internet Access Pens, Pencils, Markers, & Expo Markers Additional Internet Resources https://www.practicepython.org https://codingbat.com/python

Understandings

Students will understand the fundamentals of good programming practices in Python. They will learn how to find errors and identify common errors. This understanding of what to look for when debugging will be of great benefit as they create more complex programs. With a basic understanding of the Python language's syntax, students will have the foundation for more complex programming projects. They will also take their good programming practices (spacing, indentations, commenting) and apply them to more intricate/longer programs throughout the unit. Additionally, students will also understand the value of critical programming concepts in Python: variables, conditionals, repetition statements, functions, events, and event handlers.

ESSENTIAL QUESTIONS
What is a programming language and why are there so many?
There are many programming languages today, is it necessary to know all of them?
How is computer science used in today's world and are you capable of correcting, modifying, and creating mini programs?

Students will know…	Students will be able to…
Students will know the syntax rules of the Python Programming Language. They will know how to analyze, create, and test code. Students will know how to create programs using the appropriate compiler. In creating their programs students will see how programming can be used to create technology products to complete a variety of tasks.	*Students will be able to:* *Identify, understand, and apply different programming concepts using Python* ● *Understand the syntax rules of Python.* ● *Understand the importance of formatting: spacing, indentations, and commenting out code.*

	● *Understand how the computer takes lines of code and puts it in a format that allows it to run a program.* ● *Be able to troubleshoot issues arising due to syntax errors.* ● *Be able to create programs that output visual results.* ● *Understand how to use primitive data type variables and arithmetic expressions to make dynamic programs.* ● *Be able to follow/trace the execution of programs that use variables and arithmetic expressions.* ● *Apply concepts to create dynamic stand alone mini programs from start to finish.* ● *Create the necessary variables and algorithms in logical order.* ● *Repeat lines of code efficiently by using loops.* ● *Understand the importance of control statements in improving the dynamic nature and efficiency of their code.* ● *Create programs that work under various scenarios.* ● *Be able to identify what control statements are appropriate at given times* ○ *Decision making statements* ○ *Nested decision* ○ *Loops*

Stage 2 – Assessment Evidence

Performance Tasks: After review of PPT's and classroom discussions students will answer online review questions pertaining to programming concepts as they relate to Python. These questions will focus on generic programming terms and concepts. Questions will test the students' understanding of Python, compilers (how they allow human words to be used to command computers), basic Python syntax (rules of the language), variables, mathematical expressions, conditionals, repetition statements, and functions. Students will trace completed programs for an understanding of how code should be typed. In order to become familiar with the syntax of the Python language students will complete online tutorials and guided practice coding exercises. Students will also be required to modify code for different purposes forcing them to understand where changes need to be made. Students will be required to create programs based on given criteria. These programs will be dynamic through the use of variables, mathematical expressions, conditionals, repetition statements and functions. All output for their programs will be visual by nature.	Other Evidence **(Alternate Assessments)**: Vocabulary quizzes Chapter tests Completion of assigned worksheets/questions Classroom discussion and interaction during activities (collaboration & communication) Completion of programming projects using Python Programming Language Online mini-quizzes throughout chapter to monitor student understanding

Stage 3 – Learning Plan

This interactive unit is maintained online through the Carnegie Mellon Computer Science Academy. Each chapter will be supported by teacher-presented powerpoint, supplemental handouts and notes, mini videos explaining and illustrating concepts, and online programming exercises that will allow students the ability to apply new concepts in actual code. Within chapters students will watch tutorials, read documentation online, modify existing code to change how it works, debug code, and finally create their own original programs using Python.

Creating Drawings
- *1.1 Notes and Exercises*
- *1.2 Notes and Exercises*
- *1.3 Notes and Exercises*
- *1.4 End of Unit Exercises*
- *Creative Tasks*
- *Review/Quizzes*

Functions, Mouse Events, and Properties
- *2.1 Notes and Exercises*
- *2.2 Notes and Exercises*
- *2.3 Notes and Exercises*
- *2.4 End of Unit Exercises*
- *Creative Tasks*
- *Review/Quizzes*

Mouse Motion Events, Conditionals, and Helper Functions
- *3.1 Notes and Exercises*
- *3.2 Notes and Exercises*
- *3.3 Notes and Exercises*
- *3.4 End of Unit Exercises*
- *Creative Tasks*
- *Review/Quizzes*

More Conditionals, Key Events, and Methods
- *4.1 Notes and Exercises*
- *4.2 Notes and Exercises*
- *4.3 Notes and Exercises*
- *4.4 End of Unit Exercises*
- *Creative Tasks*
- *Review/Quizzes*

Complex Conditionals and More Key Events
- *5.1 Notes and Exercises*
- *5.2 Notes and Exercises*
- *5.3 End of Unit Exercises*
- *Creative Tasks*
- *Review/Quizzes*

Groups, Step Events, and Motion
- *6.1 Notes and Exercises*
- *6.2 Notes and Exercises*
- *6.3 Notes and Exercises*
- *6.4 End of Unit Exercises*
- *Creative Tasks*

Review/Quizzes

Local Variables, For Loops and Random Values
7.1 Notes and Exercises
7.2 Notes and Exercises
7.3 Notes and Exercises
7.4 End of Unit Exercises
Creative Tasks
Review/Quizzes

New Shapes, Types and Nested Loops
8.1 Notes and Exercises
8.2 Notes and Exercises
8.3 Notes and Exercises
8.4 End of Unit Exercises
Creative Tasks
Review/Quizzes

Strings and While Loops
9.1 Notes and Exercises
9.2 Notes and Exercises
9.3 Notes and Exercises
9.4 End of Unit Exercises
Creative Tasks
Review/Quizzes

Optional Units Below - completion of units will depend on timing. These units could be explored by students that are pacing ahead of the class.
Lists and Return Values
10.1 Notes and Exercises
10.2 Notes and Exercises
10.3 Notes and Exercises
10.4 End of Unit Exercises
Creative Tasks
Review/Quizzes

2D Lists and Board Games
11.1 Notes and Exercises
11.2 Notes and Exercises
11.3 End of Unit Exercises
Creative Tasks
Review/Quizzes

Planned Differentiation & Interventions for Tiers I, II, III, ELL, 504s, SPED, and Gift & Talented Students

• Rethink and revise. Dig deeper into ideas at issue (through the faces of understanding). Revise, rehearse, and refine, as needed. Guide students in self-assessment and self-adjustment, based on feedback from inquiry, results, and discussion.

• *Evaluate understanding. Reveal what has been understood through final performances and products. Involve students in a final self-assessment to identify remaining questions, set future goals, and point toward new units and lessons.* •*Tailor (personalize) the work to ensure maximum interest and achievement. Differentiate the approaches used and provide sufficient options and variety (without compromising goals) to make it most likely that all students will be engaged and effective.*
Gifted & Talented: Students will be directed to online websites for more challenging problems and practice (https://www.practicepython.org & https://codingbat.com/python). These websites provide mini lessons and additional programming practice/problems. Students will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code). Students will be assigned Lead Learner Role (see Tier III)
Tier I: Students will be directed to online websites for more challenging problems and practice (https://www.practicepython.org & https://codingbat.com/python). These websites provide mini lessons and additional programming practice/problems. Students will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code).
Tier II: Students will be provided more guided practice. When creating programs students will be provided comments identifying what they need to do in each section of the program. These comments will be in written form and will help with structuring their programs and identifying the needed components of the program.
Tier III: Students will be provided more one-on -one instruction. In classes where possible students will also be assigned a lead learner (student that is excelling that can provide assistance whenever necessary). These students will have differentiated tests - fill in the bank with word banks, fewer options on multiple choice exams. Students will be provided notes and access to all materials. For programming projects students will be provided the same features as Tier II but will also get starter code.
ELL: Students have access to Google translate. Students may have assignments and assessments printed in their native language if available (many of the unplugged activities have this accommodation. Students will be partnered with other students that speak their native language if possible. Students may take quizzes/exams with an ELL teacher. Additionally, ELLstudents will be grouped into appropriate tiers and receive those additional accommodations.
504s: Accommodating based on recommendations. Ability for notes and lessons to be unplugged, select grouping, etc. Additionally, 504 students will be grouped into appropriate tiers and receive those additional accommodations.
SPED: Accommodations will be provided according to IEP. Examples: preferential seating, extra time to complete assignments and quizzes, read quiz aloud, copy of notes. Additionally, SPED students will be grouped into appropriate tiers and receive those additional accommodations.

Unit 3 Object Oriented Programming: Java Syntax, Errors, and Primitive Data		
Content & Practice Standards	**Interdisciplinary Standards**	**Critical Knowledge & Skills**
CRP2. Apply appropriate academic and technical skills. CRP6. Demonstrate creativity and innovation. CRP11. Use technology to enhance productivity. 8.1.12.CS.2: Model interactions between application software, system software, and hardware. 8.1.12.CS.3: Compare the functions of application software, system software, and hardware. 9.4.12.CT.1: Identify problem-solving strategies used in the development of an innovative product or practice 9.4.12.CT.2: Explain the potential benefits of collaborating to enhance critical thinking and problem solving 8.1.12.AP.1 - Design algorithms to solve computational problems using a combination of original and existing algorithms. 8.1.12.AP.3 - Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice. 8.1.12.AP.4 - Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue. 8.2.12.E.4 - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements). 8.2.12.E.CS1 - Computational thinking and computer programming as tools used in design and engineering. ST 2.3: Apply a currently applicable computer programming language to a process, project, plan or issue as assigned. ST 2.5: Apply a technological, scientific, or mathematical concept (use of algorithms) when communicating with others on issues, plans, processes, problems, or concepts.	**RST.9-10.2.** Determine the central ideas, themes, or conclusions of a text; trace the text's explanation or depiction of a complex process, phenomenon, or concept; provide an accurate summary of the text. **RST.9-10.5.** Analyze the relationships among concepts in a text, including relationships among key terms (e.g., force, friction, reaction force, energy). **RST.11-12.2.** Determine the central ideas, themes, or conclusions of a text; summarize complex concepts, processes, or information presented in a text by paraphrasing them in simpler but still accurate terms. **A -REI** A. Understand solving equations as a process of reasoning and explain the reasoning **F-BF** A. Build a function that models a relationship between two quantities 1. Write a function that describes a relationship between two quantities	*Identify, understand, and apply different programming concepts using the Java Language* ● *Understand the syntax rules of the Java Language.* ● *Understand the importance of formatting: spacing, indentations, and commenting out code.* ● *Understand how the computer takes lines of code and puts it in a format that allows it to run a program.* ● *Be able to troubleshoot issues arising due to syntax errors.* ● *Be able to create programs that output text as well as graphics.* ● *Understand how to use primitive data type variables and arithmetic expressions to make dynamic programs.* ● *Be able to follow/trace the execution of programs that use variables and arithmetic expressions.* ● *Apply concepts to create dynamic stand alone mini programs from start to finish.* ● *Create the necessary variables and algorithms in logical order.*

12.9.3.IT-PRG.4 - Demonstrate the effective use of software development tools to develop software applications. 12.9.3.IT-PRG.5 - Apply an appropriate software development process to design a software application. 12.9.3.IT-PRG.6 - Program a computer application using the appropriate programming language.		

Unit 3 Object Oriented Programming: Java Syntax, Errors, and Primitive Data

Stage 1 – Desired Results

Unit Summary

Students will be introduced to the Java Programming Language. They will understand the history of the Java Language as well as begin to look at the syntax rules. Students will reflect on programming terminology that they have already been introduced to, understanding how the terms are applied in the Java Language. They will begin to trace and explain completed code as well as type their own stand alone mini programs. Students will be able to write programs and debug basic errors.

Core and Supplemental Materials/Resources (open resources)

Unit Resources
Textbook – Fundamentals of Java Chapter 2-3
- Exam View Assessment Suite Chapter 2-3
- Instructor Resources CD Chapter 2-3 (PPT, Handouts, Solutions)

Textbook – Java Software Solutions AP Computer Science Chapter 2
- Website – PreLab Projects, Post Chapter Questions and Projects

Planning Poster Board (mapping programs)
Mini-Whiteboards: Tracing Programs
Software/Compiler – JGrasp & BlueJ
Computers
Pens, Pencils, Markers, & Expo Markers
Internet Access

Internet Resource Links:
- http://homes.cs.washington.edu/~reges/teals/
- http://homes.cs.washington.edu/~reges/teals/bjp.shtml
- https://codehs.com/
- http://chortle.ccsu.edu/CS151/cs151java.html
- http://csunplugged.org/activities/
- https://users-mooc.amplify.com/apcs
- www.phschool.com

Understandings

Students will understand the fundamentals of good programming practices in Java. They will learn how to find errors as well as identify errors that are common. This understanding of what to look for when debugging will be of great benefit as they create more complex programs. With a basic understanding of the Java Language's syntax, students will have the foundation for more complex programming projects. They will also take their good programming practices (spacing, indentations, commenting) and apply them to more intricate/longer programs throughout the year. Additionally, students will understand the value of variables in creating dynamic programs. By the end of the unit students will have an understanding of how to create a basic program. This will serve as the foundation for all their programs moving forward.

ESSENTIAL QUESTIONS
What is a programming language and why are there so many?
What are the basics of any object oriented programming language?
There are many programming languages today, is it necessary to know all of them?
How does an understanding of object oriented programming, regardless of language, prepare you for success in Computer Science?
How is computer science used in today's world and are you capable of correcting, modifying, and creating mini programs?

Students will know…	**Students will be able to…**
Students will know the syntax rules of the Java Programming Language. They will know how to analyze, test, and create code. Students will know how to create programs using the appropriate compiler. In creating their programs students will see how programming can be used to create technology products to complete a variety of tasks.	*Students will be able to:* *Identify, understand, and apply different programming concepts using the Java Language* ● *Understand the syntax rules of the Java Language.* ● *Understand the importance of formatting: spacing, indentations, and commenting out code.* ● *Understand how the computer takes lines of code and puts it in a format that allows it to run a program.* ● *Be able to troubleshoot issues arising due to syntax errors.* ● *Be able to create programs that output text as well as graphics.* ● *Understand how to use primitive data type variables and arithmetic expressions to make dynamic programs.* ● *Be able to follow/trace the execution of programs that use variables and arithmetic expressions.* ● *Apply concepts to create dynamic stand alone mini programs from start to finish.* ● *Create the necessary variables and algorithms in logical order.*
Stage 2 – Assessment Evidence	
PERFORMANCE TASKS After review of PPT's and classroom discussions students will answer textbook review questions pertaining to programming concepts and how they align to the Java Language. These questions will focus on generic programming terms and concepts. Questions will test the students' understanding of an Object Oriented Programming Language (Java), compilers (how they allow human words to be used to command computers), basic Java syntax (rules of the language), variables, and mathematical expressions. Students will trace completed programs from the text. In order to become familiar with the syntax of the Java language students will type completed programs, but have to add comments explaining the code's execution. This will reinforce proper syntax while also forcing them to explain how programs work illustrating the importance of commenting out their code so non-programmers can understand their program's intent. Students will also be required to modify code for different purposes forcing them to understand where changes need to be made. Students will be required to create programs based on given criteria. These programs will be dynamic through the use of variables, mathematical expressions, and predefined classes that allow outsiders to interact with the program. This will be the ultimate application of material covered throughout preceding chapters.	Other Evidence **(Alternate Assessments)**: Vocabulary quizzes Chapter tests Completion of assigned worksheets/questions Classroom discussion and interaction during activities (collaboration & communication) Completion of programming projects using Java Programming Language Online mini-quizzes throughout chapter to monitor student understanding
Stage 3 – Learning Plan	
Java Syntax and Semantics – rules of the language and structure of programs *Java Chapter 2-3 Powerpoint* ● *Pre-Lab Activities from phschool.com website (pre-test: what do we know?)*	

- *Vocabulary Terms*

Tracing/Commenting out of sample programs throughout chapter to explain program's execution

End of Section and Chapter review questions throughout text in Chapters 2 & 3

Worksheets form University of Washington Computer Science & Engineering: TEALS Workshop

- *http://homes.cs.washington.edu/~reges/teals/*
  - *Videos that should be viewed in advance of classroom lessons (certain assignments – flip classroom)*
- *http://homes.cs.washington.edu/~reges/teals/bjp.shtml*
  - *Online Quizzes and Flashcards to Monitor Student progress throughout unit*
- *http://chortle.ccsu.edu/CS151/cs151java.html*

Modifying and Creating Interactive Programs with Variables

Students will create stand alone mini programs using JGrasp and BlueJ
Programming Projects Include

- *Write a program that displays your name, address, hobbies*
- *Write a program that performs simple mathematical calculations using numbers input by the user*
- *Write a program that takes as an input of number of kilometers and prints the corresponding number of miles*
- *Write a program that calculates and prints the number of minutes in a year*
- *Write a program that expects an object's mass and velocity as inputs and prints its momentum*
- *Write a program that calculates area, surface area, volume, etc. of shapes with given dimensions*
- *Write a program that calculates pay based on hours worked and hourly pay*

Additional Program Possibilities:

- *http://homes.cs.washington.edu/~reges/teals/bjp.shtml*
- *www.phschool.com*
- *Instructor Resources CD – Fundamental of Java AP Computer Science Essentials (textbook)*

Planned Differentiation & Interventions for Tiers I, II, III, ELL, 504s, SPED, and Gift & Talented Students

• Rethink and revise. Dig deeper into ideas at issue (through the faces of understanding). Revise, rehearse, and refine, as needed. Guide students in self-assessment and self-adjustment, based on feedback from inquiry, results, and discussion.

• Evaluate understanding. Reveal what has been understood through final performances and products. Involve students in a final self-assessment to identify remaining questions, set future goals, and point toward new units and lessons.

•Tailor (personalize) the work to ensure maximum interest and achievement. Differentiate the approaches used and provide sufficient options and variety (without compromising goals) to make it most likely that all students will be engaged and effective.

Gifted & Talented:
Students will be directed to online websites for more challenging problems and practice (University of Washington and Georgia Tech). These websites provide different level questions (easy, medium, hard) as well as different types (code correction, MC code analysis, and code implementation). They will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code). Students will be assigned Lead Learner Role (see Tier III)

Tier I: Students will be directed to online websites for more challenging problems and practice (University of Washington and Georgia Tech). These websites provide different level questions (easy, medium, hard) as well as different types (code correction, MC code analysis, and code implementation). They will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code).
Tier II: Students will be provided more guided practice. When creating programs students will be provided comments identifying what they need to do in each section of the program. These comments will be in written form and will help with structuring their programs and identifying the needed components of the program.
Tier III: Students will be provided more one-on -one instruction. In classes where possible students will also be assigned a lead learner (student that is excelling that can provide assistance whenever necessary). These students will have differentiated tests - fill in the bank with word banks, fewer options on multiple choice exams. Students will be provided notes and access to all materials. For programming projects students will be provided the same features as Tier II but will also get starter code.
ELL: Students have access to Google translate. Students may have assignments and assessments printed in their native language if available (many of the unplugged activities have this accommodation). Students will be partnered with other students that speak their native language if possible. Students may take quizzes/exams with an ELL teacher. Additionally, ELL students will be grouped into appropriate tiers and receive those additional accommodations.
504s: Accommodating based on recommendations. Ability for notes and lessons to be unplugged, select grouping, etc. Additionally, 504 students will be grouped into appropriate tiers and receive those additional accommodations.
SPED: Accommodations will be provided according to IEP. Examples: preferential seating, extra time to complete assignments and quizzes, read quiz aloud, copy of notes. Additionally, SPED students will be grouped into appropriate tiers and receive those additional accommodations.

Unit 4 Control Statements: Conditionals, If, If-Else, & Loops		
Content & Practice Standards	**Suggested Standards for ELA Practice**	**Critical Knowledge & Skills**
CRP2. Apply appropriate academic and technical skills. CRP6. Demonstrate creativity and innovation. CRP11. Use technology to enhance productivity. 8.1.12.CS.2: Model interactions between application software, system software, and hardware. 8.1.12.CS.3: Compare the functions of application software, system software, and hardware. 9.4.12.CT.1: Identify problem-solving strategies used in the development of an innovative product or practice 9.4.12.CT.2: Explain the potential benefits of collaborating to enhance critical thinking and problem solving	**RST.9-10.2.** Determine the central ideas, themes, or conclusions of a text; trace the text's explanation or depiction of a complex process, phenomenon, or concept; provide an accurate summary of the text. **RST.9-10.5.** Analyze the relationships among concepts in a text, including relationships among key terms (e.g., force, friction, reaction force, energy). **RST.11-12.2.** Determine the central ideas, themes, or conclusions of a text; summarize complex concepts, processes, or information presented in a text by paraphrasing them in simpler but still accurate terms. **A -REI**	*Identify, understand, and apply Control Statements* ● *Understand conditionals and how they execute based on Boolean logic.* ● *Understand how to create complex, multi-faceted conditions.* ● *Understand/Follow Decision Trees.* ● *Understand the difference between selection statements and repetition statements.* ● *Understand how to follow the execution order of control statements in other programmers' code.* ● *Trace code using conditionals and loops.* ● *Make decisions in their programs based on user input.* ● *Repeat lines of code efficiently by using loops.*

8.1.12.AP.1 - Design algorithms to solve computational problems using a combination of original and existing algorithms.
8.1.12.AP.3 - Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice.
8.1.12.AP.4 - Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue.
 	
8.2.12.E.4 - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements).
8.2.12.E.CS1 - Computational thinking and computer programming as tools used in design and engineering.
 	
ST 2.3: Apply a currently applicable computer programming language to a process, project, plan or issue as assigned.
ST 2.5: Apply a technological, scientific, or mathematical concept (use of algorithms) when communicating with others on issues, plans, processes, problems, or concepts.
 	
12.9.3.IT-PRG.4 - Demonstrate the effective use of software development tools to develop software applications.
12.9.3.IT-PRG.5 - Apply an appropriate software development process to design a software application.
12.9.3.IT-PRG.6 - Program a computer application using the appropriate programming language.
	A. Understand solving equations as a process of reasoning and explain the reasoning
	
	F-BF
	A. Build a function that models a relationship between two quantities
	1. Write a function that describes a relationship between two quantities
		• *Understand the importance of control statements in improving the dynamic nature and efficiency of their code.*
		• *Create programs that work under various scenarios.*
		• *Be able to identify what control statements are appropriate at given times*
		  ○ *Decision making statements*
		  ○ *Nested decision*
		  ○ *Loops*

Unit 4 Control Statements: Conditionals, If, If-Else, & Loops

Stage 1 – Desired Results

Unit Summary	Core and Supplemental Materials/Resources (open resources)
Students will explore the use of conditionals and loops in programming granting programmers more control of the execution of their programs. They will understand Boolean logic and conditionals in determining a program's order of execution. This will make their programs more versatile and allow them to work differently under unique scenarios. Students will be able to follow the execution of other's code using conditionals and loops as well as implement these concepts in their programs.	Unit Resources Textbook – Fundamentals of Java Chapter 4 & 7 • Exam View Assessment Suite Chapter 4 & 7 • Instructor Resources CD Chapter 4 & 7 (PPT, Handouts, Solutions) Textbook – Java Software Solutions AP Computer Science Chapter 3 • Website – PreLab Projects, Post Chapter Questions and Projects Planning Poster Board (mapping programs) Mini-Whiteboards: Tracing Programs Computers: Software/Compiler – JGrasp & BlueJ

	Pens, Pencils, Markers, & Expo Markers Internet Resource Links: ● http://homes.cs.washington.edu/~reges/teals/ ● http://homes.cs.washington.edu/~reges/teals/bjp.shtml ● https://codehs.com/ ● http://chortle.ccsu.edu/CS151/cs151java.html ● http://csunplugged.org/activities/ ● https://users-mooc.amplify.com/apcs ● www.phschool.com

UNDERSTANDINGS

Students will be able to follow Boolean logic in programs understanding that order of execution is not always top to bottom. They can gain control and improve efficiency in their programs by including this feature. Students will also be able to use their programs in slightly different ways due to implementation of conditionals allowing the skipping of certain code when necessary. Once again they will value the ability to type multi-purpose programs rather than several mini stand alone programs that do the same thing with just one minor difference. Efficiency is essential in technology when we consider the cost of storage capacity and processing power. It is better to create efficient code than redundant code and excessive mini-programs that will create unnecessary overhead.

ESSENTIAL QUESTIONS
Why is efficiency important when using technology and when programming?
Why is succinct and dynamic coding important?
Why and under what scenarios would programmers want to maintain control of their programs even after code development?
How can we create effective code that works under several scenarios and processes through multiple times without being repetitive?
How does an understanding of object oriented programming, regardless of language, prepare you for success in Computer Science?
How is computer science used in today's world and are you capable of correcting, modifying, and creating mini programs?

Students will know…	**Students will be able to…**
Students will know how to analyze, test, and create code that use higher level programming concepts. They will know the importance of efficiency in programming including how to repeat code and make decisions within their programs execution pattern. They will know the many ways programmers can control their programs execution with a focus on response to user input.	*Students will be able to:* *Identify, understand, and apply Control Statements* ● *Understand conditionals and how they execute based on Boolean logic* ● *Understand how to create complex, multi-faceted conditions* ● *Understand/Follow Decision Trees* ● *Understand the difference between selection statements and repetition statements* ● *Understand how to follow the execution order of control statements in other programmers' code* ● *Trace code using conditionals and loops* ● *Make decisions in their programs based on user input* ● *Repeat lines of code efficiently by using loops* ● *Understand the importance of control statements in improving the dynamic nature and efficiency of their code* ● *Create programs that work under various scenarios* ● *Be able to identify what control statements are appropriate at given times* ○ *Decision making statements* ○ *Nested decision* ○ *Loops*

Stage 2 – Assessment Evidence

Performance Tasks:	Other Evidence **(Alternate Assessments)**:
After review of PPT's and classroom discussions students will answer textbook review questions pertaining to control statements: conditions, If, If/Else, Loops, and nested statements. Questions will test the students understanding of control statements requiring them to evaluate these statements to make sure they are using the correct syntax, make sure the control statements work correctly based on intent, and be able to trace the logic of control statements to determine what the execution of the program will be. *Students will type completed programs, adding comments explaining the code's execution. This will reinforce proper syntax while also forcing them to explain how programs work. This will illustrate that students can evaluate conditions and proceed to the next appropriate piece of code demonstrating an understanding of how conditionals work. They will also be required to modify code for different purposes forcing them to understand where changes need to be made.* *Students will be required to create programs that implement conditional statements whether this be selection statements or repetition statements. These programs will be dynamic and cumulative, requiring students to not only implement control statements but also continue to use variables, mathematical expressions, and allow user input. This will be the ultimate application of material covered throughout the preceding chapters.*	*Vocabulary quizzes* *Chapter tests* *Completion of assigned worksheets/questions* *Classroom discussion and interaction during activities* *Completion of programming projects using Java Language* *Online mini-quizzes throughout chapter to monitor student understanding*

Stage 3 – Learning Plan

Java Syntax and Semantics – rules and structure for conditional statements, selection statements (if, if/else, nested if/else), and repetition statements (loops: while & for and the nested version of each)

Fundamentals of Java: Chapter 4 & 7 Powerpoint

- *Pre-Lab Activities from phschool.com website (pre-test: what do we know?)*
- *Vocabulary Terms*

Tracing/Commenting out of sample programs throughout chapter to explain program's execution

Fundamentals of Java: End of Section and Chapter review questions throughout text in Chapters 4 & 7

Java Software Solutions for AP Computer Science A: end of chapter review questions (Chapter 3)

Worksheets form University of Washington Computer Science & Engineering: TEALS Workshop

- *http://homes.cs.washington.edu/~reges/teals/*

Videos that should be viewed in advance of classroom lessons (certain assignments – flip classroom)

- *http://homes.cs.washington.edu/~reges/teals/bjp.shtml*

Online Quizzes and Flashcards to Monitor Student progress throughout unit

- *http://chortle.ccsu.edu/CS151/cs151java.html*

Online practice questions – codehs.org

Modifying and Creating Interactive Programs with Variables and User Input

Students will create stand alone mini programs using JGrasp & BlueJ
Programming Projects Include:

- *Program that calculates a students GPA (no weighting)*
- *Program that calculates a students GPA considering each classes' level and weighting*
- *Program that takes a number input by a user and sums all the evens between 2 and that number verifying that the number the user put in is greater than 2*
- *Program that calculates your pay based on hourly wage and overtime*
- *Program that determines Credit Card Charges, Balances, and Minimum Payments*
- *Program a High Low Guessing Game*
- *Program a Rock Paper Scissors Game*
- *Program a Slot Machine*

Additional Program Possibilities:

- *http://homes.cs.washington.edu/~reges/teals/bjp.shtml*
- *www.phschool.com*
- *Instructor Resources CD – Fundamental of Java AP Computer Science Essentials (textbook)*

Planned Differentiation & Interventions for Tiers I, II, III, ELL, 504s, SPED, and Gift & Talented Students

• Rethink and revise. Dig deeper into ideas at issue (through the faces of understanding). Revise, rehearse, and refine, as needed. Guide students in self-assessment and self-adjustment, based on feedback from inquiry, results, and discussion.

• Evaluate understanding. Reveal what has been understood through final performances and products. Involve students in a final self-assessment to identify remaining questions, set future goals, and point toward new units and lessons.

•Tailor (personalize) the work to ensure maximum interest and achievement. Differentiate the approaches used and provide sufficient options and variety (without compromising goals) to make it most likely that all students will be engaged and effective.

Gifted & Talented:
Students will be directed to online websites for more challenging problems and practice (University of Washington and Georgia Tech). These websites provide different level questions (easy, medium, hard) as well as different types (code correction, MC code analysis, and code implementation). They will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code). Students will be assigned Lead Learner Role (see Tier III)

Tier I:
Students will be directed to online websites for more challenging problems and practice (University of Washington and Georgia Tech). These websites provide different level questions (easy, medium, hard) as well as different types (code correction, MC code analysis, and code implementation). They will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code).

Tier II:
Students will be provided more guided practice. When creating programs students will be provided comments identifying what they need to do in each section of the program. These comments will be in written form and will help with structuring their programs and identifying the needed components of the program.

Tier III:

Students will be provided more one-on -one instruction. In classes where possible students will also be assigned a lead learner (student that is excelling that can provide assistance whenever necessary). These students will have differentiated tests - fill in the bank with word banks, fewer options on multiple choice exams. Students will be provided notes and access to all materials. For programming projects students will be provided the same features as Tier II but will also get starter code.
ELL: Students have access to Google translate. Students may have assignments and assessments printed in their native language if available (many of the unplugged activities have this accommodation). Students will be partnered with other students that speak their native language if possible. Students may take quizzes/exams with an ELL teacher. Additionally, ELL students will be grouped into appropriate tiers and receive those additional accommodations.
504s: Accommodating based on recommendations. Ability for notes and lessons to be unplugged, select grouping, etc. Additionally, 504 students will be grouped into appropriate tiers and receive those additional accommodations.
SPED: Accommodations will be provided according to IEP. Examples: preferential seating, extra time to complete assignments and quizzes, read quiz aloud, copy of notes. Additionally, SPED students will be grouped into appropriate tiers and receive those additional accommodations.

Unit 5 Writing Classes		
Content & Practice Standards	**Suggested Standards for ELA Practice**	**Critical Knowledge & Skills**

CRP2. Apply appropriate academic and technical skills. CRP6. Demonstrate creativity and innovation. CRP11. Use technology to enhance productivity. 8.1.12.CS.2: Model interactions between application software, system software, and hardware. 8.1.12.CS.3: Compare the functions of application software, system software, and hardware. 9.4.12.CT.1: Identify problem-solving strategies used in the development of an innovative product or practice 9.4.12.CT.2: Explain the potential benefits of collaborating to enhance critical thinking and problem solving 8.1.12.AP.1 - Design algorithms to solve computational problems using a combination of original and existing algorithms. 8.1.12.AP.3 - Select and combine control structures for a specific application based upon performance and readability, and identify trade-offs to justify the choice. 8.1.12.AP.4 - Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue. 8.2.12.E.4 - Use appropriate terms in conversation (e.g., troubleshooting, peripherals, diagnostic software, GUI, abstraction, variables, data types and conditional statements). 8.2.12.E.CS1 - Computational thinking and computer programming as tools used in design and engineering. ST 2.3: Apply a currently applicable computer programming language to a process, project, plan or issue as assigned. ST 2.5: Apply a technological, scientific, or mathematical concept (use of algorithms) when communicating with others on issues, plans, processes, problems, or concepts. 12.9.3.IT-PRG.4 - Demonstrate the effective use of software development tools to develop software applications. 12.9.3.IT-PRG.5 - Apply an appropriate software development process to design a software application.	**RST.9-10.2.** Determine the central ideas, themes, or conclusions of a text; trace the text's explanation or depiction of a complex process, phenomenon, or concept; provide an accurate summary of the text. **RST.9-10.5.** Analyze the relationships among concepts in a text, including relationships among key terms (e.g., force, friction, reaction force, energy). **RST.11-12.2.** Determine the central ideas, themes, or conclusions of a text; summarize complex concepts, processes, or information presented in a text by paraphrasing them in simpler but still accurate terms. **A -REI** A. Understand solving equations as a process of reasoning and explain the reasoning **F-BF** A. Build a function that models a relationship between two quantities 1. Write a function that describes a relationship between two quantities	*Understand and Create Classes* ● *Understand the value of creating classes even though they do not run.* ● *Understand Classes as blueprints that other programs will interact with.* ● *Understand the components of Classes: Data Declarations and Method Declarations.* ● *Understand the structure of a class.* ● *Be able to trace the execution of driver programs using different classes, including being able to follow the jumping back and forth between classes.* ● *Be able to create their own classes creating the necessary encapsulated data and methods accessible by outsiders.* ● *Understand scope in terms of what can and can't be accessed and used in their classes.* ● *Understand how to call constructors to create objects from classes and then use those objects they have created to access methods.* ● *Understand and implement overloaded and overridden methods by evaluating method signatures.* ● *Create methods with correct method headers (return types and parameters) as well as method bodies that act almost like mini programs.* ● *Create driver programs that illustrate how their classes work.*

12.9.3.IT-PRG.6 - Program a computer application using the appropriate programming language.		

Unit 5 Control Statements: Writing Classes

Stage 1 – Desired Results

Unit Summary	Core and Supplemental Materials/Resources (open resources)
With an understanding of objects and object oriented programming students will start to better organize their code by creating their own classes to define objects that perform services they define. This unit will focus on the details of class definitions including the structure and semantics of methods and the scope and encapsulation of data. Students will understand the value of creating classes which means defining methods once and then calling these methods with objects created from that classes. They will understand classes as a blueprint and how to create and interact with objects from this blueprint.	Unit Resources Textbook – Fundamentals of Java Chapter 5-6 ● Exam View Assessment Suite Chapter 5-6 ● Instructor Resources CD Chapter 5-6 (PPT, Handouts, Solutions) Textbook – Java Software Solutions AP Computer Science Chapter 4 ● Website – PreLab Projects, Post Chapter Questions and Projects Planning Poster Board (mapping programs) Mini-Whiteboards: Tracing Programs Computers: Software/Compiler – JGrasp & BlueJ Pens, Pencils, Markers, & Expo Markers Internet Resource Links: ● http://homes.cs.washington.edu/~reges/teals/ ● http://homes.cs.washington.edu/~reges/teals/bjp.shtml ● https://codehs.com/ ● http://chortle.ccsu.edu/CS151/cs151java.html ● http://csunplugged.org/activities/ ● https://users-mooc.amplify.com/apcs ● www.phschool.com

Understandings

Writing classes is at the foundation of Object Oriented Programming. Up to this point we have used other programmers' classes for the purpose they created them for. Now we will be able to create our own classes from which we and others can develop objects and interact with our code. This improves our efficiency by limiting redundant code. It is also what object oriented programming is all about. This is the beginning process of illustrating how open source programming works – allowing outsiders to use your code, but controlling the scenarios in which they can.

ESSENTIAL QUESTIONS
What is object oriented programming and how does it allow for better organized and understandable code?
Understanding the role of programming/technology in today's world, why is organization and standard structure necessary when programming?
What is a class in Java and although it does not run, why do we write them?
How can we create effective code that works under several scenarios?
How does an understanding of object oriented programming, regardless of language, prepare you for success in Computer Science?
How is computer science used in today's world and are you capable of correcting, modifying, and creating mini programs?

Students will know…	**Students will be able to…**
Students will know the importance of collaboration and code reuse in programming. They will know how to better organize their code so that commonly used functions and classes can be shared amongst multiple programs. Students will know how to encapsulate information and control what outsiders can access. They	*Students will be able to:* ● *Understand and Create Classes* ● *Understand the value of creating classes even though they do not run* ● *Understand Classes as blueprints that other programs will interact with* ● *Understand the components of Classes: Data Declarations and Method Declarations*

will know that programming should be open source but some things must be protected to maintain the integrity of their programs.	• *Understand the structure of a class* • *Be able to trace the execution of driver programs using different classes, including being able to follow the jumping back and forth between classes* • *Be able to create their own classes creating the necessary encapsulated data and methods accessible by outsiders* • *Understand scope in terms of what can and can't be accessed and used in their classes* • *Understand how to call constructors to create objects from classes and then use those objects they have created to access methods* • *Understand and implement overloaded and overridden methods by evaluating method signatures* • *Create methods with correct method headers (return types and parameters) as well as method bodies that act almost like mini programs* • *Create driver programs that illustrate how their classes work*
Stage 2 – Assessment Evidence	
Performance Tasks: *After review of PPT's and classroom discussions students will answer textbook review questions pertaining to Writing Classes: components – data declarations & method declarations, encapsulation, classes as a blueprint, developing driver programs that use a written class, using constructors, using methods with parameters, format of methods in classes (create proper headers and method body). Questions will test the students' understanding of classes whether it is developing them, tracing execution, or interacting correctly with them.* *Students will trace completed programs from the text. In order to become familiar with the syntax of Classes in Java and practice mapping out the execution, students will type completed programs (both driver program and class itself), adding comments explaining the code's execution. This will reinforce proper syntax while also forcing them to explain how programs work together jumping back and forth between their code. This will illustrate that students can jump between code, returning to the correct position when finished.* *Students will be required to create classes and driver programs that use them. They will have to demonstrate they know the proper structure of classes as well as which things should be accessible and which should not be. They will also show the ability to interact with classes that they have created as well as classes others have created. In creating their classes, in particular methods, students will be taking concepts covered throughout the year that had been put into mini programs, and efficiently putting them into methods. Students will use material covered throughout preceding chapters in formulating their classes.*	Other Evidence **(Alternate Assessments)**: *Vocabulary quizzes* *Chapter tests* *Completion of assigned worksheets/questions* *Classroom discussion and interaction during activities* *Completion of programming projects using Java Language* *Online mini-quizzes throughout chapter to monitor student understanding*
Stage 3 – Learning Plan	
Java Syntax and Semantics – rules and structure for Classes: data declarations and method declarations *Fundamentals of Java: Chapter 5-6 Powerpoint*	

- *Pre-Lab Activities from phschool.com website (pre-test: what do we know?)*
- *Vocabulary Terms*
- *Tracing/Commenting out of sample programs throughout chapter to explain program's*
- *execution*
- *Fundamentals of Java: End of Section and Chapter review questions throughout text in Chapters 5-6*

Java Software Solutions for AP Computer Science A: end of chapter review questions (Chapter 4)

Worksheets form University of Washington Computer Science & Engineering: TEALS Workshop

- *http://homes.cs.washington.edu/~reges/teals/*

Videos that should be viewed in advance of classroom lessons (certain assignments – flip classroom)

- *http://homes.cs.washington.edu/~reges/teals/bjp.shtml*

Online Quizzes and Flashcards to Monitor Student progress throughout unit

- *http://chortle.ccsu.edu/CS151/cs151java.html*

Online practice questions – codehs.org

Modifying and Creating Interactive Programs with Variables and User Input

Students will create classes using BlueJ/JGrasp software

- *Programming Projects Include:*
  - *Create a class that represents a Student: class should include name, address,and grade level variables defined in the constructor, a method that returns the students grade level, a method that calculates and returns the students gpa (accepts 1 parament, # of classes), and a toString method that returns the students name, grade level, and GPA. Also create a driver program that demonstrates how the student class works.*
  - *Create a class that represents a BankAccount: an account should have an owner's name, account number, and starting balance, a constructor that allows a client to supply the name, account number, and starting balance, a method for depositing money, a method for withdrawing money, and a toString method that returns the owner's name, account number, and balance. Also create a driver program that demonstrates how the BankAccount class works.*
  - *Stick Figure Movie Project – applet that works like a flip book and appears to be moving Create Stick Figure Class with various methods providing the stick figure with actions it can perform. Update a driver program to call the methods you created in the stick figure class. This program is what will run the animation*

Additional Program Possibilities:

- *http://homes.cs.washington.edu/~reges/teals/bjp.shtml*
- *www.phschool.com*
- *Instructor Resources CD – Fundamental of Java AP Computer Science Essentials (textbook)*

Planned Differentiation & Interventions for Tiers I, II, III, ELL, 504s, SPED, and Gift & Talented Students

• Rethink and revise. Dig deeper into ideas at issue (through the faces of understanding). Revise, rehearse, and refine, as needed. Guide students in self-assessment and self-adjustment, based on feedback from inquiry, results, and discussion.

• Evaluate understanding. Reveal what has been understood through final performances and products. Involve students in a final self-assessment to identify remaining questions, set future goals, and point toward new units and lessons.

•Tailor (personalize) the work to ensure maximum interest and achievement. Differentiate the approaches used and provide sufficient options and variety (without compromising goals) to make it most likely that all students will be engaged and effective.
Gifted & Talented: Students will be directed to online websites for more challenging problems and practice (University of Washington and Georgia Tech). These websites provide different level questions (easy, medium, hard) as well as different types (code correction, MC code analysis, and code implementation). They will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code). Students will be assigned Lead Learner Role (see Tier III)
Tier I: Students will be directed to online websites for more challenging problems and practice (University of Washington and Georgia Tech). These websites provide different level questions (easy, medium, hard) as well as different types (code correction, MC code analysis, and code implementation). They will get immediate results from these websites. Students will add complexity to their programs (user input, more complex coding practice, higher expectations in regards to formatting output and commenting code).
Tier II: Students will be provided more guided practice. When creating programs students will be provided comments identifying what they need to do in each section of the program. These comments will be in written form and will help with structuring their programs and identifying the needed components of the program.
Tier III: Students will be provided more one-on-one instruction. In classes where possible students will also be assigned a lead learner (student that is excelling that can provide assistance whenever necessary). These students will have differentiated tests - fill in the bank with word banks, fewer options on multiple choice exams. Students will be provided notes and access to all materials. For programming projects students will be provided the same features as Tier II but will also get starter code.
ELL: Students have access to Google translate. Students may have assignments and assessments printed in their native language if available (many of the unplugged activities have this accommodation). Students will be partnered with other students that speak their native language if possible. Students may take quizzes/exams with an ELL teacher. Additionally, ELL students will be grouped into appropriate tiers and receive those additional accommodations.
504s: Accommodating based on recommendations. Ability for notes and lessons to be unplugged, select grouping, etc. Additionally, 504 students will be grouped into appropriate tiers and receive those additional accommodations.
SPED: Accommodations will be provided according to IEP. Examples: preferential seating, extra time to complete assignments and quizzes, read quiz aloud, copy of notes. Additionally, SPED students will be grouped into appropriate tiers and receive those additional accommodations.