Date: 03/17/2021 **High School**____District________

El Monte Union High School District
Course Outline

Course Title: *Intro to Systems Programming*

Textbook(s): *QBasic Fundamentals and Style with an Introduction to Microsoft Visual Basic; Python for Everyone*

Copyright date/Edition: 2001 and 2016

Transitional*_______(Eng. Dept. Only)

Sheltered (SDAIE)*____Bilingual*___

AP**________Honors**___________

Department: CTE

CTE***:
Industry Sector: Information Technology

Pathway: Software and Systems Design
(check one)
_X__Intro ___ Intermediate
___Capstone ____Elective

Grade Level (s): 9, 10

Semester___________Year__X_____

Year of State Framework Adoption_2017

This course meets graduation requirements:

( ) English
( ) Fine Arts
( ) Foreign Language
( ) Health & Safety
( ) Math
( ) Physical Education
( ) Science
( ) Social Science
( ) Elective

This course meets a-g requirements:
(_____) "a" – Social Studies
(_____) "b" – ELA
(_____) "c" – Math
(_____) "d" – Lab Science
(_____) "e" – Language (not English)
(_____) "f" – Vis/Perf Arts
(_____) "g" – College prep Elective

Department/Cluster Approval	Date

Is this course an adaptation from another source?
(__X___) No
(_____) Yes
If yes, please indicate the source of the original course:
____

*Instructional materials appropriate for English Language Learners are required.

**For AP/Honors course attach a page describing how this course is above and beyond a regular course. Also, explain why this course is the equivalent of a college level class.

***For CTE, attach the CTE course outline created in the online template (http://ctecourse.scoe.net/).

1. Prerequisite(s): None

2. **Short description of course which may also be used in the registration manual:**
   - **Objectives of course**
     Refer to the comprehension and application objectives in the unit descriptions
   - **3-5 sentences explaining overall course content**
     This course provides an in-depth introduction to coding in Python .Upon completion, students will master fundamental coding concepts such as statements, variables, expressions, conditionals, and loops. Students will also gain proficiency with advanced topics including software libraries, automation, and sprite-based graphics. In addition, students will learn how to plan and track the progress of large coding projects, debug errors, and improve the readability of their code as well as learn industry practices such as pair programming, code reviews, and role-based project development. Throughout the course, students will continuously demonstrate their knowledge through both traditional assessments and coding projects such as games, animations, and other interactive programs, and real-world coding projects, crafting the foundations of their professional portfolio.
   - **Indicate references to state framework(s)/standards (If state standard is not applicable then national standards should be used) + Student performance standards**
     <u>Industry Sector Anchor Standards:</u> Information and Communication Technologies
     2.0 Communications
     2.3 Interpret verbal and nonverbal communications and respond appropriately.
     2.4 Demonstrate elements of written and electronic communication such as accurate spelling, grammar, and format.
     2.6 Advocate and practice safe, legal, and responsible use of digital media information and communications technologies.
     2.7 Use technical writing and communication skills to work effectively with diverse groups of people.
     2.8 Understand the principles of a customer-oriented service approach to users.
     3.0 Career Planning and Management
     3.3 Explore how information and communication technologies are used in career planning and decision making.
     3.4 Research the scope of career opportunities available and the requirements for education, training, certification, and licensure.
     3.5 Integrate changing employment trends, societal needs, and economic conditions into career planning.
     4.0 Technology
     4.1 Use electronic reference materials to gather information and produce products and services.
     4.2 Employ technology-based communications responsibly and effectively to explore complex systems and issues.
     4.3 Use information and communication technologies to synthesize, summarize, compare, and contrast information from multiple sources.
     5.0 Problem Solving and Critical Thinking
     5.1 Identify and ask significant questions that clarify various points of view to solve problems.
     5.2 Solve predictable and unpredictable work-related problems using various types of reasoning (inductive, deductive) as appropriate.

5.3 Use systems thinking to analyze how various components interact with each other to produce outcomes in a complex work environment.
5.4 Interpret information and draw conclusions, based on the best analysis, to make informed decisions.
5.5 Use a logical and structured approach to isolate and identify the source of problems and to resolve problems.
5.6 Know the available resources for identifying and resolving problems.
5.7 Work out problems iteratively and recursively.
5.8 Create and use algorithms and solve problems.
5.9 Deconstruct large problems into components to solve.
5.10 Use multiple layers of abstraction.
5.12 Apply the concepts of Boolean logic to decision making and searching.
6.0 Health and Safety
6.2 Interpret policies, procedures, and regulations for the workplace environment, including employer and employee responsibilities.
6.3 Use health and safety practices for storing, cleaning, and maintaining tools, equipment, and supplies.
6.4 Practice personal safety when lifting, bending, or moving equipment and supplies.
6.5 Demonstrate how to prevent and respond to work-related accidents or injuries; this includes demonstrating an understanding of ergonomics.
6.6 Maintain a safe and healthful working environment.
6.8 Maintain a safe and healthful working environment.
7.0 Responsibility and Flexibility
7.2 Explain the importance of accountability and responsibility in fulfilling personal, community, and workplace roles.
7.3 Understand the need to adapt to changing and varied roles and responsibilities.
7.4 Practice time management and efficiency to fulfill responsibilities.
7.5 Apply high-quality techniques to product or presentation design and development.
7.7 Demonstrate the qualities and behaviors that constitute a positive and professional work demeanor, including appropriate attire for the profession.
7.8 Explore issues of global significance and document the impact on the Information and Communication Technologies sector.
8.0 Ethics and Legal Responsibilities
8.4 Explain the importance of personal integrity, confidentiality, and ethical behavior in the workplace.
9.0 Leadership and Teamwork
9.2 Identify the characteristics of successful teams, including leadership, cooperation, collaboration, and effective decision-making skills as applied in groups, teams, and career technical student organization activities.
9.3 Understand the characteristics and benefits of teamwork, leadership, and citizenship in the school, community, and workplace setting.
9.6 Respect individual and cultural differences and recognize the importance of diversity in the workplace.
9.7 Participate in interactive teamwork to solve real Information and Communication Technologies sector issues and problems.
10.0 Technical and Knowledge Skills
10.1 Interpret and explain terminology and practices specific to the Information and Communication Technologies sector.
10.3 Construct projects and products specific to the Information and Communication Technologies sector requirements and expectations.
10.10 Manage files in a hierarchical system.
11.0 Demonstration and Application

11.1 Utilize work-based/workplace learning experiences to demonstrate and expand upon knowledge and skills gained during classroom instruction and laboratory practices specific to the Information and Communication Technologies sector program of study.
11.2 Demonstrate proficiency in a career technical pathway that leads to certification, licensure, and/or continued learning at the postsecondary level.
11.5 Create a portfolio, or similar collection of work, that offers evidence through assessment and evaluation of skills and knowledge competency as contained in the anchor standards, pathway standards, and performance indicators.

Career Pathway Standards: **C. Software and Systems Development Pathway**
C1.0 Identify and apply the systems development process.
C1.3 Identify and describe how specifications and requirements are developed for new and existing software applications.
C1.4 Work as a member of, and within the scope and boundaries of, a development project team.
C1.5 Track development project milestones using the concept of versions.
C2.0 Define and analyze systems and software requirements.
C2.1 Describe the major purposes and benefits of development, including automation, improving productivity, modeling and analysis, and entertainment.
C2.2 Recognize and prevent unintended consequences of development work: programming errors, security issues, health and environmental risks, and privacy concerns.
C2.3 Develop strategies that target the specific needs and desires of the customer.
C2.4 Analyze customers' needs for development.
C2.5 Determine and document the requirements and alternative solutions to fulfill the customers' needs.
C4.0 Develop software using programming languages.
C4.1 Identify and describe the abstraction level of programming languages from low-level, hardware-based languages to high-level, interpreted, Web-based languages.
C4.2 Describe the interaction and integration of programming languages and protocols such as how client-side programming can work with server-side programming to use a query language to access a database.
C4.3 Identify and use different authoring tools and integrated development environments (IDEs).
C4.4 Identify and apply data types and encoding.
C4.5 Demonstrate awareness of various programming paradigms, including procedural, object-oriented, event-driven, and multithreaded programming.
C4.6 Use proper programming language syntax.
C4.7 Use various data structures, arrays, objects, files, and databases.
C4.8 Use object-oriented programming concepts, properties, methods, and inheritance.
C4.9 Create programs using control structures, procedures, functions, parameters, variables, error recovery, and recursion.
C4.10 Create and know the comparative advantages of various queue, sorting, and searching algorithms.
C4.11 Document development work for various audiences, such as comments for other programmers, and manuals for users.
C5.0 Test, debug, and improve software development work.
C5.1 Identify the characteristics of reliable, effective, and efficient products.
C5.3 Use strategies to optimize code for improved performance.
C5.4 Test software and projects.
C5.5 Evaluate results against initial requirements.
C7.0 Develop Web and online projects.
C7.1 Identify the hardware (server) and software required for Web hosting and other services.
C8.0 Develop databases.

C8.5 Use queries to extract and manipulate data (select queries, action queries).
C9.0 Develop software for a variety of devices, including robotics.
C9.3 Use hardware to gain input, process information, and take action.
C9.4 Apply the concepts of embedded programming, including digital logic, machine-level representation of data, and memory-system organization.

Key Academic Standards from the Academic Alignment Matrix
Language Standards: 11-12.1, 11.12.2
Reading Standards: 11-12.3, 11-12.7
Writing Standards: 11-12.2, 11-12.4, 11-12.6, 11-12.8

CSTA Standards
3A-AP-13: Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
3A-AP-15: Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
3A-AP-16: Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
3A-AP-17: Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
3A-IC-26: Demonstrate ways a given algorithm applies to problems across disciplines.
3B-AP-10: Use and adapt classic algorithms to solve computational problems.
3B-AP-15: Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.
3B-DA-07: Evaluate the ability of models and simulations to test and support the refinement of hypotheses.

- **Evaluation/assessment/rubrics**
  - Kahoot quizzes
  - Check-For-Understanding
  - Projects
  - Algorithm and Abstractions description assignments
  - Informal observations of individual and group tasks

- **Include minimal attainment for student to pass course**
  Grade > 70%

Coding Concepts

- Text Input and Output
- Statements
- Expressions
- Variables
- Mathematical Operators
- Conditionals
- Booleans
- Logical Operators
- While Loops
- Libraries
- Randomness

- Debugging
- Coordinates
- Windows
- Drawing Lines and Shapes
- RGB Colors
- Tuples
- Procedural Animation
- Event Loops
- Mouse and Keyboard Input
- Timing and Framerate
- Lists
- Indexes
- For-Each Loops
- For-Range Loops
- Sprite Images
- Spritesheet Animation
- Collision
- Writing Functions
- Arguments vs Parameters
- Return Values
- Default Parameters
- Passing by Reference

Student Outcomes (TechSmart)

- Write programs that make computers follow instructions
- Write code that makes decisions, choosing between multiple options
- Write code that loops, repeating instructions until certain outcomes are reached
- Pull in outside libraries that increase the capabilities of their programs
- Create code that opens windows on a computer and draw graphics with shapes and colors
- Animate shapes using traditional frame-based animation techniques
- Interpret signals from the mouse and keyboard to control their programs
- Store and organize multiple items at once using a list data structure
- Analyze and manipulate lists with looping code
- Create code image objects from .jpg and .png images
- Load animations from a sheet of frame images
- Analyze when two image objects are colliding on screen and write code that reacts to it
- Organize their code to be more efficient and useful
- Use functions to write multiple sections of code that communicate with each other

<u>Students will:</u>

- Develop coding fundamentals.
- Develop independent and collaborative strategies that will help them communicate around computing as they learn and reinforce the fundamental concepts of coding.
- Use collaborative strategies that coding professionals use when creating programs and applications.
- Understand that computer science can be more than just innovation and creative expression; it can be powerful in trying to solve many problems in today's world.
- Apply an Agile development process and task decomposition to solve a problem that meets the needs of

others.

- Gain insight on the importance of creativity, persistence, and value of diverse perspectives in an iterative development process.
- Understand lower levels of abstraction in a programming language.
- Explore image processing and other innovations that are changing our society 9. Investigate the wide range of careers in computer science and how computational thinking is an important part of professions today and in the future.
- Learn how to take collaborations to scale to achieve a common goal.
- Implement algorithms using conditionals and loops in Python.
- Create a game simulation and reinforce what they have learned about functions, arguments, and return values.
- Learn about model abstraction and the impact that simulation and data are having across career fields.
- Apply their Python skills to compete in a rock-paper-scissors game, developing functions to implement a complex strategy that attempts to detect and react to their opponent's strategy.
- Be exposed to common web programming languages as they learn to make decisions about how people view and interact with a website and how to protect data that is exchanged.
- Collaborate the way computing professionals do as they pursue solutions to authentic needs.

3. Course content:
Number of units (minimum of 6): ______8_______

INSTRUCTINAL UNITS

Unit 1	**Topic:** Linear Programming

Equipment and Health

Student Objectives:

Students will:

- Understand the importance of keeping digital tools clean.
- Properly retrieve digital devices.
- Practice ergonomics.

Lesson 1.1 Statements and Variables

Comprehension Objectives:

- Define the lesson terms
- Describe how a computer executes code
- Identify input and output in a program
- Identify variables and their values
- Identify, describe, and differentiate between camel Case notation and underscore notation for variable names

Application Objectives:

- Use basic console text input and output commands
- Store values in variables
- Combine strings (both variables and literals) using the + operator
- Debug common problems related to lesson topics

Lesson 1.2 Libraries

Comprehension Objectives:

- Define the lesson terms

- Describe what happens in the flow of code when a function is called
- Identify function calls in code
- Identify the arguments sent to a function
- Explain why arguments may be necessary for functions
- Use documentation to identify what arguments are necessary for a given function

Application Objectives:

- Import a library
- Call a function from a library using the correct arguments
- Debug common problems related to lesson topics

Lesson 1.3 Values

Comprehension Objectives:

- Define the lesson terms
- Identify which data type is most appropriate for a given situation
- Distinguish between literals and variables
- Identify the data type of a given value
- Identify when a basic mathematical operator will produce an integer and when it will produce a float
- Give the order of operations for basic mathematical operators

Application Objectives:

- Combine strings (both variables and literals) using the + operator
- Use typecasting to temporarily alter the type of a value
- Use basic mathematical operators on integers and floats: +, -, /, *
- Create a printable string value by combining strings and numbers
- Debug common problems related to lesson topics

Research Question: Tech Impact

Comprehension Objectives:

- Define the lesson terms
- Give examples of search strategies that could be used to research the lesson topic
- Explain why citing sources is important
- Learn about careers that use computer science
- Describe how technology has changed culture over time

Application Objectives:

- Gather information from a variety of sources
- Evaluate the accuracy and bias of sources
- Provide citations for sources used
- Evaluate how technology has impacted various career fields
- Consider how technology might impact a career field in the future

Lesson 1.4 Expressions

Comprehension Objectives:

- Define the lesson terms
- Given a statement that uses a compound assignment operator, give the full version of the statement with separate assignment and math operators and vice-versa
- List the benefits of using compound assignment operators
- Identify an expression within a line of code
- Distinguish between a statement (performs a complete action) and an expression (produces a value, but does nothing with it)
- Identify when a command (such as input) is both an expression and a statement
- Identify when a function may be treated as an expression (e.g. when it returns a value)

- Describe what happens in the flow of code when a function with a return value is called
- Use documentation to identify whether a function returns a value that may be stored
- Identify 'None' as the value returned by any function that does not have an explicit return value

Application Objectives:

- Typecast an input expression to produce a number result
- Predict the result of an expression
- Use compound assignment operators (+=. -=. *=,\=)
- Describe where these operators fall in the order of operations for Python
- Store the result of a function
- Use a function as part of an expression
- Use the following techniques: Typecasting Input, Incrementing a String
- Debug common problems related to lesson topics

Industry Practice: Planning a Program

Comprehension Objectives:

- Define the lesson terms
- Describe the product life cycle as a formal process for creating software
- Identify and describe "Envision" and "Design" as the first two steps in the product life cycle
- Explain why a planning phase is necessary and useful before beginning a larger project
- Differentiate between pseudocode and true syntax

Application Objectives:

- Outline a given program as pseudocode
- Break a given program / problem down into smaller features / sub-problems
- Brainstorm ideas and present the result as a list of detailed features
- Translate a feature list into a pseudocode outline
- Implement a program from a pseudocode outline

Unit 2	**Topic:** Decisions

Lesson 2.1 Conditionals (if)

Comprehension Objectives:

- Define the lesson terms
- Identify if statements in code
- Identify the condition within an if statement
- Describe how an if statement makes a decision
- Identify which comparison operator is most appropriate in a given context
- Describe where comparison operators fall in the order of operations for Python
- Differentiate between the "=" and "==" operators and describe the function of each
- Explain how whitespace is used to delineate the beginning and end of conditional sections

Application Objectives:

- Write an if statement to make a decision
- Predict which code within a conditional will execute from looking at a program
- Use the following techniques: User Choice, Running Total, Limit Number
- Debug common problems related to lesson topics

Lesson 2.2 Conditionals (elif and else)

Comprehension Objectives:

- Define the lesson terms

- Describe the flow of a conditional with elif- and else-clauses
- Describe the general format of a clause (i.e. begins with a keyword and ends with a ':')
- Identify when a conditional structure is nested
- Identify the range described by the 'min < num < max' chained comparison format

Application Objectives:

- Predict which code within a conditional will execute from looking at a program
- Predict which conditionals within a nested structure will execute from looking at a program
- Write a conditional to make a decision between multiple cases
- Determine whether a number falls into a range defined by the min < num < max format
- Debug common problems related to lesson topics

Industry Practice: Code Style

Comprehension Objectives:

- Define the lesson terms
- Describe the benefits of good code style and commenting
- Identify "PEP-8" as the commonly accepted Python code style guidelines
- Explain that code style does not affect program output or functionality
- Differentiate between standard and header
- comment syntax

Application Objectives:

- Improve the readability of programs using an good code style
- Improve the readability of programs using an appropriate level of comment density

Research Question: Automation

Comprehension Objectives:

- Describe how technology has changed culture over time

Application Objectives:

- Make predictions about future technology based on existing technology
- Evaluate how technology has impacted various career fields

Industry Practice: Debugging

Comprehension Objectives:

- Define the lesson terms
- Identify the line number within an error message
- Give examples of scenarios where line numbers may not be accurate (e.g. missing parentheses)
- Give an approximate plain English translation of an error message
- Identify and describe two different code-debugging strategies:
  - Using print statements
  - Reading the error message

Application Objectives:

- Choose which debugging strategy is most effective for a given situation
- Use both strategies to find and fix errors

2.3: Built-In Libraries

Comprehension Objectives:

- Define the lesson terms
- Identify and describe the random and math libraries
- Give examples of commands found in the random and math libraries
- Explain how to use documentation to find a full list of available commands in a library
- Give examples of how randomness may be used in a program
- Explain the relationship between randomness and Artificial Intelligence

- List common programming uses for the modulus operator

Application Objectives:

- Use the random.randint() function
- Use the modulus operator (%)
- Use the exponentiation operator (**)
- Use the floor division operator (//)
- Use advanced math operations from the math library (such as sqrt)
- Find and use other operations from math and random without explicit introduction to them
- Use the following techniques: Random Choice, Weighted Choice
- Debug common errors related to the lesson topics

2.4: Booleans

Comprehension Objectives:

- Define the lesson terms
- Contrast between the logical operators
- Identify boolean expressions in code
- Recognize comparison operators as operators that produce booleans
- Explain that booleans can be stored in variables
- like other data types
- Differentiate between well-formatted boolean variable conditions and redundant (bad boolean zen) versions

Application Objectives:

- Choose which logical operator is appropriate to combine values in a given situation
- Predict the values that will result from given boolean expressions
- Given a set of constraints or conditions under which something will happen, translate this into a compound boolean expression
- Simply a complex compound boolean expression by replacing various expressions with variables
- Reduce a bad boolean zen condition to a simpler form
- Use a single boolean variable as a condition
- Use the following techniques: Flexible Input, Reduce Compound Expressions
- Debug common problems related to lesson topics

Industry Practice: Scoping and Presenting Work

Comprehension Objectives:

- Define the lesson terms
- Explain why proper scoping for a project is important
- Differentiate between a modular approach and other, more monolithic approaches
- Explain how a modular approach allows for scoping up or down

Application Objectives:

- Choose a project of reasonable scope for a given time-frame
- Create a meaningful presentation of a program, explaining points of interest
- Present failures as well as successes as a normal part of a retrospective
- Divide a project into multiple releases or versions

Continue meeting Equipment and Health objectives per Unit 1

Unit 3	**Topic:** Loops

3.1: While Loops

Comprehension Objectives:

- Define the lesson terms
- Describe the logical flow of a loop
- Explain the importance of changing the loop condition inside the loop (e.g. avoiding infinite loops)
- List benefits of using loops (simplify code, run until signalled to stop, etc.)

Application Objectives:

- Looking at a loop, determine how much it will repeat / when it will stop
- Use while loops to repeat code until the user chooses to stop
- Create loops that are governed by a single boolean control variable
- Use the following techniques: Force Correct Input, Nested Loops, Player Turns, True Until False
- Debug common problems related to lesson topics

3.2: Controlling Loops

Comprehension Objectives:

- Define the lesson terms
- List and describe different variations on while loops (while, loop else clause)
- Differentiate between loops that end normally and loops that end with break
- Differentiate between the effect of 'break' and 'continue' within a loop block
- Describe alternatives to using a 'continue' statement (e.g. using conditionals to decide whether to do part of the loop block)
- Identify situations where it would be reasonable to use break

Application Objectives:

- Use break to exit a loop early
- Use continue to skip skip the remainder of a loop iteration
- Debug common problems related to lesson topics

3.3: Classes

Comprehension Objectives:

- Define the lesson terms
- Describe how an instance is related to a class
- Describe how methods and attributes are related to a class
- Give examples of classes

Application Objectives:

- Set and get fields on an instance
- Call methods of an instance
- Use documentation to get information about the attributes and methods of a class without prior instruction on them
- Use the technique: Change an Instance With a Function
- Debug common problems related to lesson topics

3.4: Graphics

Comprehension Objectives:

- Define the lesson terms
- Identify the arguments required to create various visual objects (window, sprite, etc.)
- Describe how the main loop is used to keep a program open
- Describe the conditions necessary to open, update, and close a window (e.g. the main loop

using the is_running field, and the window.finish_frame command)
- Identify what happens when you forget the window.finish_frame command (e.g. infinite loop)
- Describe how coordinates are used to represent a position on-screen
- List and describe different text alignments (left, center, right)
- Contrast the programming coordinate space with the math coordinate space
- Explain the value of labeling constants
- Contrast variables and constants
- Identify a constant based on the style conventions of its name (e.g. in ALL_CAPS)

Application Objectives:
- Open a window using tsapp
- Draw static sprites with tsapp
- Draw text to screen with tsapp
- Given a set of coordinates and a window size, roughly estimate the coordinate position
- Use the following techniques: Precise Positioning, Draw Order
- Debug common problems related to lesson topics

Research Question: Intellectual Property

Comprehension Objectives:
- Explain, compare, and debate the effects of intellectual property laws
- Explain the necessity of providing attribution, and the effects of failing to do so

Application Objectives:
- Debate laws and regulations that impact the development and use of software
- Compare and evaluate licenses for different types of usage, including code licenses

3.5: Animation

Comprehension Objectives:
- Define the lesson terms
- Describe how animation occurs because of rapid change in each iteration of a loop
- Describe a sprite's speed as the number of pixels it moves in one second
- Compare and contrast x vs y speeds
- Compare and contrast positive vs negative speeds
- Describe how a spritesheet is transformed into an animated image
- Contrast between movement-based animation and image-based animation

Application Objectives:
- Perform simple animations by moving objects in a loop
- Use an animated sprite based on a spritesheet
- Animate a change in a sprite by manually changing the image
- Change the speed of a sprite's visual or movement-based animation
- Use the following techniques: Change Animation Rate, Change Direction
- Debug common problems related to lesson topics

3.6: Interaction

Comprehension Objectives:
- Define the lesson terms
- Explain the difference between states (e.g. whether a button is down) and events (e.g whether a button was pressed on this frame)

Application Objectives:
- Call methods that check for the current state of keys and mouse (e.g. position, is_down)
- Call methods that check for events related to keys and mouse (e.g. was_pressed)
- Call methods that check for collision between mouse and sprite, or two sprites

- Use the following techniques: Follow Mouse, Move with Arrow Keys
- Calculate duration by subtracting two points in time
- Assign multiple variables at once using comma syntax
- Debug common problems related to lesson topics

3.7: For-Range Loops

Comprehension Objectives:

- Define the lesson terms
- Differentiate between while and for loops
- Describe how the value of the loop variable changes as the loop continues
- Describe how any for-range loop could be written as a while loop
- Give the default ‘range’ values when not overridden (e.g. ‘0’ for start and ‘1’ for step)

Application Objectives:

- Choose whether a for-loop or a while-loop is more appropriate for a given situation
- Use a for-range loop to loop a specific number of times
- Use for-range variations to count by amounts other than 1
- Use for-range variations to count backwards
- Use continue, break and else with a for loop
- Use the following techniques: Counting Down, Row of Sprites
- Debug common problems related to lesson topics

Industry Practice: Kanban

Comprehension Objectives:

- Define the lesson terms
- Explain the importance of tracking work for large projects
- Describe how kanban is used to track work
- Compare kanban to other simple forms of work-tracking, such as checklists

Application Objectives:

- Create a kanban board that tracks tasks in a large project
- Choose appropriate lanes for the type of project being undertaken
- Debug common problems related to lesson topics

Continue meeting Equipment and Health objectives per Unit 1

Unit 4	**Topic:** Lists and For-Each Loops

Lists 4.1: Lists and For-Each Loops

Comprehension Objectives

- Define a list as a changeable collection of ordered information
- Identify the valid and invalid indexes in a list
- Describe the properties of elements contained in a list
- Describe how the loop variable of a for-each loop changes when iterating over a list
- Compare and contrast different types of loops

Application Objectives

- Create a new list of one or more elements
- Iterate over the elements in a list using a loop
- Retrieve the length of a list
- Reference an element in a list by its index

4.2: List Operations

Comprehension Objectives

- Give examples of problems that can be solved by querying a list
- Give examples of how a list can change during the execution of a program
- Determine the most appropriate method to add or remove elements from a list
- Describe how a new list can be created by analyzing the elements of an existing list
- Describe problems caused by adding or removing an item from a list while iterating over that list

Application Objectives

- Query a list about the presence, number, or location of an element
- Add or remove an element from a list
- Map a list to another list using a loop
- Filter a list into another list using a loop

Research Project: Private Data

Comprehension Objectives

- Describe tradeoffs between allowing information to be public and keeping it private and secure
- Describe ways in which data can be collected that might not be obvious to the data's subject
- Give examples of malicious or controversial usage of private data

Application Objectives

- Debate laws and regulations around data privacy
- Gather information from a variety of sources
- Evaluate the accuracy and bias of sources
- Provide citations for sources used

4.3: Advanced List Operations

Comprehension Objectives

- Give examples of when to use a sorted list over an unsorted list
- Describe the benefits of using randomness with collections
- Explain the difference between equality and identity in lists
- Compare and contrast the use of + and * operators on lists with their use on other data types

Application Objectives

- Change the order of elements in a list by sorting and reversing
- Find the minimum or maximum value in a list
- Apply functions from the random library to lists
- Check whether two variables refer to the same list or two lists with equal values
- Concatenate and repeat lists

4.4: Strings as Collections

Comprehension Objectives

- Compare and contrast strings, as a collection of characters, to lists
- Identify which collection operations can and cannot be used on strings
- Identify which elements of an existing collection will be selected by a slice operation
- Identify the default values for collection slicing when no values are given
- Recognize that slicing a collection with a negative step value results in a new reversed collection

Application Objectives

- Use previously explored collection operations on strings and substrings
- Create a new collection by slicing an existing collection

4.5: String Operations

Comprehension Objectives

- Identify and differentiate between spaces, tabs and newlines
- Describe how an escape sequence makes it easier to display certain characters
- Identify when a list of substrings is preferred over a string of the same characters and vice versa

- Choose which string operation is most appropriate for a given task
- Explain why an intermediate step is needed to swap two existing values

Application Objectives

- Create a string with one or more escape sequences
- Convert a string to a list and vice versa
- Create a new string by replacing all occurrences of a substring with a new substring
- Convert the letters of a string to the same case
- Check whether a string contains only the same type of characters

Industry Practice: Pair Coding

Comprehension Objectives

- Define pair coding as a process where two people write the same lines of code together
- Describe the how professionals work together when pair coding
- Explain the benefits and challenges of pair coding

Application Objectives

- Build a program with another student by pair coding as both a driver and a navigator

Unit 5	**Topic:** Data Structures

5.1: Tuples

Comprehension Objectives

- Define a tuple as an unchangeable collection of ordered information
- Explain why it can be useful to use a tuple instead of a list
- Identify which collection operations can and cannot be used on tuples
- Identify which data types a collection can be casted to or from
- Predict which values will be assigned to each variable when unpacking a tuple

Application Objectives

- Create a new tuple of one or more elements
- Use previously explored collection operations on tuples
- Cast a collection from one data type to another
- Assign the elements of a collection to one or more variables in a single unpacking statement

5.2: Dictionaries

Comprehension Objectives

- Define a dictionary as a changeable collection of unordered key-value pairs
- Describe the properties of key-value pairs in dictionaries
- Given a set of data, choose which collection type would be most appropriate for storing it
- Identify which dictionary operation is most appropriate to solve a given problem

Application Objectives

- Create a new dictionary of zero or more key-value pairs
- Reference a value in a dictionary by a key
- Use previously explored collection operations on dictionaries
- Update a dictionary with the contents of another dictionary
- Iterate over a dictionary using a for-each loop
- Retrieve a list of the keys, values, or items in a dictionary

Research Project: The Internet

Comprehension Objectives

- Identify systems that use networks and provide real-world examples

Application Objectives

- Model how information is stored as packets
- Model how packets are communicated across a network

5.3: Nested Collections

Comprehension Objectives

- Differentiate between the way lists and dictionaries organize nested collections of data
- Identify that collections can be nested to any depth
- Determine which nested collections would best model a given set of data
- Describe how sequential bracket notation is used to reference data in a nested collection
- Describe the algorithm to translate pixel coordinates from a screen into grid coordinates in a nested list of lists and vice versa

Application Objectives

- Create a new heterogeneous nested collection
- Create a new homogeneous nested collection
- Use sequential bracket notation to reference data in a nested collection
- Use iteration to build a nested collection and to reference its data

5.4: Web APIs

Comprehension Objectives

- Define the lesson terms
- List some common Python web APIs
- List some pros and cons of using web APIs versus direct web-scraping
- Discuss the value of collecting and analyzing data
- Compare different data visualizations, and argue about which is most effective

Application Objectives

- Use a web API to interface with data on a website
- Debug common problems related to lesson topics

Industry Practice: Code Review

Comprehension Objectives

- Define the lesson terms
- Describe the process of code reviews
- List some benefits of code reviews

Application Objectives

- Review code with other students
- Revise code based on code reviews

Unit 6	**Topic:** Functions

6.1: Custom Functions

Comprehension Objectives

- Define the lesson terms
- List some benefits of putting code in a function
- Describe the difference between a parameter and an argument
- Describe recursion and explain why it can be dangerous
- Describe the concept of "scope"
- Explain how an argument comes to take the place of a parameter

Application Objectives

- Create custom functions

- Predict when a variable will be accessed based on its scope
- Call custom functions
- Debug common problems related to lesson topics

6.2: Return Values

Comprehension Objectives

- Define the lesson terms
- Identify whether a function returns a value, and the type of that value
- Explain when and why return values are useful

Application Objectives

- Store a function's return value in a variable
- Use a function as or within an expression
- Return multiple values at once using a collection
- Debug common problems related to lesson topics

Research Project: The Internet

Comprehension Objectives

- Explain, compare, and debate the effects of intellectual property laws
- Explain the necessity of providing attribution, and the effects of failing to do so

Application Objectives

- Debate laws and regulations that impact the development and use of software
- Compare and evaluate licenses for different types of usage, including code licenses

6.3: Complex Parameters

Comprehension Objectives

- Define the lesson terms
- Differentiate between values that can and cannot be changed after being passed to a function (passed by reference vs passed by value)
- Identify which values will be used for which parameters when a function is called
- Distinguish between parameters that do and do not have default values
- Distinguish between return values and side effects

Application Objectives

- Create a function with default parameters
- Debug common problems related to lesson topics

6.4: Cryptography

Comprehension Objectives

- Define the lesson terms
- Describe how encryption protects data
- Compare different forms of cryptography across history
- List some common forms of digital data security
- List some non-digital forms of data security

Application Objectives

- Encrypt data before transfer
- Decrypt data after transfer
- Estimate the relative security of different methods of encryption and transmission
- Debug common problems related to lesson topics

Industry Practice: Software Teams

Comprehension Objectives

- Define the lesson terms
- List common roles on a software team
- List the benefits of splitting coding work amongst team members

Application Objectives

- Choose teammates to take different roles
- Support teammates in varying roles
- Complete work as part of a software team with specific roles
- Debug common problems related to lesson topics

Instructional Strategies:

- Pair Programming
- Chunk and Chew lectures
- Gamification
- Peer-to-peer assignment and assistants

Assessments:

- Kahoot quizzes
- Check-For-Understanding
- Projects

Instructional Materials:

- TechSmart (Instructor online account) ● Laptops
- Internet Access (Wifi and Ethernet access

- Algorithm and Abstractions description assignments
- Informal observations of individual and group tasks

Standards and Certifications

Upon completion of CS201: Coding with Python 1 and CS202: Coding with Python 2, students will be prepared to take the Certified Entry-Level Python Programmer (PCEP) certification exam. Additionally, students who complete CS201, CS202, and CS203: Coding with Python 3 will be prepared to take the Microsoft Technology Associate (MTA): Introduction to Programming Using Python certification exam. (TechSmart)

4. Describe how this course integrates the schools SLO (former ESLRs- Expected School-wide Learning Results):

5. Describe the Integrated ELD teaching techniques to be used to meet the needs of English Language Learners:

SDAIE strategies are expectations in all content strategies

6. Describe the interdepartmental articulation process for this course:

7. Describe how this course will integrate academic and vocational concepts, possibly through connecting activities. Describe how this course will address work-based learning/school to career concepts:

Refer to course description, standards, and unit descriptions in previous pages

8. Supplemental Materials of Instruction (Note: Materials of instruction for English Language Learners are required and should be listed below)

Type of material (book, manual, periodical, article, website, primary source document, etc.)	Author	Publisher	Edition/Year	URL	Primary book, read in its entirety? (Y/N)
QBasic fundamentals and Style with an Introduction to Microsoft Visual Basic	James S. Quasney, John Maniotes, Roy O. Foreman		Second Edition		Y
Python for Everyone	Cay S. Horstmann, Rance D. Necaise	Wiley	2016		
TechSmart					Online