ACALANES UNION HIGH SCHOOL DISTRICT
COURSE OF STUDY: CURRICULAR AREA – MATHEMATICS

COURSE TITLE:	Introduction to Computer Science
GRADE LEVEL:	10-12
COURSE LENGTH:	Semester
PREFERRED PREVIOUS OF STUDY:	Algebra 1
CREDIT:	5 credits
UC/CSU CREDIT:	Pending UC/CSU credit for mathematics requirement; subject area ("c")
GRADUATION REQUIREMENT:	Fulfills 5 units of mathematics credit (2 semesters beyond Algebra 1) required for graduation
STANDARDS AND BENCHMARKS:	California Common Core State Standards Mathematical Practices
ADOPTED:	TBD
INSTRUCTIONAL MATERIALS:	TBD

COURSE DESCRIPTION: This course exposes students to the fundamental principles and concepts of computer science and programming. Students will learn to code through hands-on assignments which include game design, lab completion, independent projects and pair programming. This course serves as both an introduction to, and foundation for, further study in computer science.

COURSE OBJECTIVES: Upon completion of the course, students will:

1. Create programs in various programming languages.
2. Use various computing techniques and strategies to complete problems.
3. Understand and appropriately apply basic data types, control structures and effectively use their methods.
4. Abstract, design, implement, debug and test programs based on various specifications.
5. Collaborate and communicate their understanding/reasoning regarding computer science principles.

ASSESSMENT: Students will be assessed through summative and formative assessments which may include:

1. Peer Evaluation of individual and group projects
2. Teacher evaluations of:
    a. Individual and group projects
    b. Labs
    c. Quizzes
    d. Tests
    e. Presentations

Assessments should be measuring the following claims:

Claim #1 – Concepts & Procedures
Students can explain and apply computer science concepts and interpret and carry out computer science procedural concepts with precision and fluency.

Claim #2 – Problem Solving
Students can solve a range of complex well-posed problems, making productive use of knowledge and problem solving strategies. These strategies include, but are not limited to: method decomposition, iterative process and program development and error analysis.

Claim #3 – Communicating Reasoning
Students can clearly and precisely construct viable arguments to support their own program design and to evaluate others' programs.

Claim #4 – Modeling and Data Analysis
Students can analyze complex, real-world scenarios and can construct and use mathematical models to interpret and solve problems. Further, the can implement using a programming language.

GRADING GUIDELINES: See AUHSD Grading Guidelines: Final Mark Rubric and Final Course mark Determination Components

COURSE CONTENT:	Mathematical Practices The Standards for Mathematical Practice are "habits of the mind of mathematically proficient students". They describe the attributes that mathematics educators at all levels are striving to develop in their students, as these practices are based on key mathematical processes and proficiencies. The goal of implementing these practices is to develop students who can use their knowledge of mathematics in flexible, sophisticated, and relevant ways across multiple disciplines. #1 Make sense of problems and persevere in solving them (Hypothesize & Strategize) • Students are: o Making conjectures about what the problem is asking and how they can begin to solve it o Checking for the reasonableness of the strategy as the work progresses and making adjustments as needed • Teachers develop this skill by having students: o Explain the meaning of the problem and/or restate the problem o Analyze the given information and develop entry points into the problem and develop strategies for solving the problem o Execute and evaluate multiple strategies #2 Reason abstractly and quantitatively (De/Contextualize) • Students are: o Determining what numbers and symbols represent through the use of diagrams, graphs or equations • Teachers develop this skill by having students: o Move between multiple representations to determine the meaning behind quantities o Express purely mathematical expressions with real world context and taking quantities out of context and representing them as abstract mathematical ideas or expressions #3 Construct viable arguments; critique others' reasoning • Students are: o Justifying their thinking by providing evidence based on mathematical properties and using that evidence to evaluate the reasoning of others • Teachers develop this skill by having students:

- o Make conjectures, compare and contrast methods, and identify flawed logic by providing counter-example

#4 Model with Mathematics

- Students are:
  - o Interpreting and constructing graphs, tables, number lines, diagrams or equations to model real-world situational data
- Teachers develop this skill by having students:
  - o Use models to make interpolative and extrapolative inferences
  - o Examine the effectiveness and appropriateness of a model

#5 Use appropriate tools strategically

- Students are:
  - o Selecting appropriate math tools and technology to help solve problems including manipulatives, graphing utilities, tables, [matrices], computer applications, compasses, etc.
- Teachers develop this skill by having students:
  - o Identify the strengths and weaknesses of different tools in relation to solving a given problem and also use tools to explore, confirm or deepen understanding

#6 Attend to Precision

- Students are:
  - o Calculating quantities accurately through proper rounding (based on context), labeling of units of measure, and checking their work
  - o Selecting a problem solving method that allows for appropriate precision
- Teachers develop this skill by having students:
  - o Formulate precise explanations of their work using vocabulary and justify their rounding process
  - o Re-examine their work or thinking process, and then demonstrate the method by which they check their answers

#7 Look For and Make Use of Structure

- Students are:
  - o Looking for patterns or relationships and using that structure to simplify complex ideas
- Teachers develop this skill by having students:

	o Extend prior knowledge of similar situations to novel ones or break down complex problems in smaller parts which resemble simpler, more familiar ideas #8 Look for and express regularity in repeated reasoning (Generalize) • Students are: o Developing general methods, rules, or short cuts and determining when they are appropriate • Teachers develop this skill by: o Facilitating activities which allow for students' "aha!" moments and then helping them use it to develop "rules" based on repeated trials with a process

Course Standards:	1. Understand programming environments, structure and execution. 2. Understand and use mathematical operations in programs and their order of execution (including +, -, /, *, modulo). 3. Understand data types and operations on different data types (For example, operations on strings or ints). 4. Understand variables including, variable assignment, modification and scope. 5. Understand conditional statements involving boolean operators. Further students will evaluate and use nested conditionals and compound conditional statements. 6. Understand use of, and creation of functions to solve problems and increase program flexibility and code reusability. Further, students will use and understand the purpose of parameters. 7. Understand and use various iterative processes including for-loops and while loops and embedded loops. Students will understand that loops increase code efficiency. 8. Understand various data structures including lists. Students will use operations on lists including insertion, deletion and referencing list values. 9. Evaluation programs for error analysis including syntax, runtime and logic errors. Students will be able to debug their own and other student's work. 10. Utilize event driven programming within simple Graphical User Interfaces (GUI). This will include use of controls structures such buttons, input boxes and utilize a canvas for visual feedback. 11. Use method decomposition to manage program complexity and readability. 12. Create large scale programs, both individually and collaboratively and present to both peers and teachers.
Scope and Sequence:	Currently, the primary languages used are Scratch and App Lab (Javascript). However, as technology advances, programming languages will be evaluated as to their appropriateness. Week #1: Scratch Lab #1-Basics Week #2- Scratch Lab #2-Loops and Repetition Week #3: Scratch lab #3-Variables Week #4: Scratch Lab #4-Conditionals Week #5: Scratch Lab #5-Music Video Week #6-7: Scratch Lab #6-Blocks and Lists Week #7-8: Scratch Lab #7-Modeling and Game Design Week #9: Robot Olympics/Project Week #10: App Lab #9 & 10-Overview Week #11: App Lab #11-Text Adventure (multi-screen) Week #12: App Lab #12-Whack-A-Mole (Lists)

	Week #13: App Lab #13-Calculator (Conditionals) Week #14: App Lab #14-Project Week #15: Final Project Week #16: Final Project Week #17: Final Project