Perris Union High School District
Course of Study

A. COURSE INFORMATION

Course Title:
Computer Science Discoveries A
- ☐ New
- ✓ Revised

If revised previous course name if changed
Computer Science Discoveries

Transcript Course Code/Number:
(To be assigned by Educational Services)

Subject Area:
- ☐ Social Science
- ☐ English
- ☐ Mathematics
- ☐ Laboratory Science
- ☐ World Languages
- ☐ Visual or Performing Arts
- ✓ College Prep Elective
- ☐ Other

Is this classified as a Career Technical Education course?
- ✓ Yes
- ☐ No

Grade Level
- ✓ MS
- ☐ HS
- ☐ 5
- ☐ 6
- ✓ 7
- ✓ 8
- ☐ 9
- ☐ 10
- ☐ 11
- ☐ 12

Required for Graduation:
- ☐ Yes
- ✓ No

Credential Required to teach this course:
Math, Business, Introductory Computer Science, Industrial Technology Ed, Computer Concepts + Application
To be completed by Human Resources only.

Signature — Date: 1-9-19

Meets UC/CSU Requirements?
- ✓ Yes
- ☐ No

Was this course *previously approved by UC* for PUHSD?
- ✓ Yes
- ☐ No

(Will be verified by Ed Services)

Meets "Honors" Requirements?
- ☐ Yes
- ✓ No

Meets "AP" Requirements?
- ☐ Yes
- ✓ No

Unit Value/Length of Course:
- ☐ 0.5 (half year or semester equivalent)
- ✓ 1.0 (one year equivalent)
- ☐ 2.0 (two year equivalent)
- ☐ Other:

Submitted by: Erik Anderson
Site: Pinacate Middle School
Date: 12/14/18

Approvals	Name/Signature	Date
Director of Curriculum & Instruction		1/9/19
Asst. Superintendent of Educational Services		1-11-19
Governing Board		

Prerequisite(s) (REQUIRED):
None
Corequisite(s) (REQUIRED):
None
Brief Course Description (REQUIRED):
Computer Science Discoveries A takes the first half of the year-long Code.org curriculum and presents it to students in a way that encourages more in-depth exploration of important computer science principles. This course will synchronize with Computer Science Discoveries B (the other half of the Code.org curriculum) to create a full computer science pathway at the middle school level. Students will gain exposure and attain proficiency in coding - in multiple languages like Python and Apps Script - physical computing, web development, portfolio management, and project architecture. Students will be empowered to create and manage authentic digital artifacts and engage with computer science as a medium for creativity, communication, problem-solving- and fun! By the end of the course, students will have used fundamental programming constructs to create personal web sites, develop animations and games, control interactions between code and electronic circuitry, and to think of the evolving digital world and their place in it.

B. COURSE CONTENT
Course Purpose (REQUIRED): *What is the purpose of this course? Please provide a brief description of the goals and expected outcomes. Note: More specificity than a simple recitation of the State Standards is needed.*
Computer Science Discoveries A is designed to introduce students to a wide range of topics in the field of computer science, largely through self-paced exploration. Using flexible and scalable curriculum content, students gather understanding not only of specific tools and computing languages, but also, more importantly, why and how certain tools are better suited to solve particular problems. The goal of Computer Science Discoveries A is to develop in students the computational thinking practices of algorithm development, problem solving and programming within the context of problems that are relevant and interesting to them.
Course Outline (REQUIRED): *Detailed description of topics covered. All historical knowledge is expected to be empirically based, give examples. Show examples of how the text is incorporated into the topics covered.*
Unit 1: Problem Solving and Computers

In this introductory unit, students are introduced to CS as a problem solving discipline, and they begin to use a formal problem solving process, a more general version of the cyclical systems development process. They apply this process to various problems, then deepen their understanding of computers as devices that solve information problems through the input, output, storage, and processing of information. The advantages and limitations to computers are explored as students compare the ways that humans and computers approach problems. At the end of the unit, students use their problem solving process to design an app that meets users' needs through the input, output, storage and processing of information.

Unit 2: Web Development

In the Web Development unit, students engage in both the design and implementation of web sites. Students first study existing websites from a design perspective, determining the features of high-quality web pages. They then design and implement their own pages using HTML and CSS, reinforcing core practices of attention to precision, abstraction, persistence, and identifying patterns and structure, and how these structures must be expressed mathematically through code. Before publishing their sites, students explore how sharing information online can compromise privacy and develop guidelines for online behavior. Other topics such as RGB (additive) color mixing, intellectual property concerns, and search engine optimization are also covered.

Unit 3: Programming Games and Animations

In the Games and Animation unit, students get their first experience with programming through the lens of creating programmatic images, animations, interactive art, and games. Early in the unit students draw images by programmatically placing geometric shapes on a coordinate plane.

Students are then gradually introduced to more complex programming constructs allowing them to bring their animations to life with movement and user-interaction. In order to manage the complexity of larger projects students learn to use high-level programming abstractions and a software design process. Along the way students investigate the mathematical relationships between position, speed, and acceleration, and how these interact with each other when objects collide. Students learn fundamental programming concepts such as variables, iteration, conditionals, and Boolean expressions. They use the systems development process to design, test, and iterate, and come to see failure and debugging as an expected and valuable part of the programming process.

Supplementary Units

Digital Art

In the digital art unit, students study geometric design and create original works of art with online software. Specifically, students analyze tessellations and frieze patterns, break them down into their constituent shapes, and recreate the full patterns on their Chromebooks. The results are typically fascinating, and students end the unit with valuable additions to their portfolios.

Physical Computing Students begin the physical computing unit with a study of basic electronic circuitry. They complete a course of study online, with a simulator, and gradually work their way to simulating LED circuits that are created with a breadboard and a microcontroller, and which are controlled by code. Future Digital World In the Future World unit, students think about the digital world and its possible future evolution. They watch several videos that examine digital frontiers in fields like medicine, machine learning, avatars, security, and big data. Students complete a group project wherein they invent and develop an idea that has not yet been invented. They conduct research to ensure that it is a new idea, and develop a business plan to create their invention and market it to appropriate audiences, and finally present their work to the class.
Writing Assignments (REQUIRED): *Give examples of the writing assignments and the use of critical analysis within the writing assignments.*
See Key Assignments

INSTRUCTIONAL MATERIALS (REQUIRED)	
Textbook #1	
Title:	Edition:
Author:	ISBN:
Publisher:	Publication Date:
Usage: ❑ Primary Text ❑ Read in entirety or near	
Textbook #2	
Title:	Edition:
Author:	ISBN:
Publisher:	Publication Date:
Usage: ❑ Primary Text ❑ Read in entirety or near	

Supplemental Instructional Materials *Please include online, and open source resources if any.*	
Estimated costs for classroom materials and supplies (REQUIRED). *Please describe in detail.* If more space is needed than what is provided, please attach backup as applicable.	
Cost for class set of textbooks: $	Description of Additional Costs:
Additional costs:$	
Total cost per class set of instructional materials:	$

Key Assignments (REQUIRED):
Please provide a detailed description of the Key Assignments including tests, and quizzes, which should incorporate not only short answers but essay questions also. How do assignments incorporate topics? Include all major assessments that students will be required to complete
Unit 1: Problem Solving and Computers Key Assignment: Students identify and define a real world problem and brainstorm ways an app could be used to help solve the problem. They then fill out a project guide identifying the inputs, outputs, storage and processing needed, as well as create a sketch of the interface. Students then give each other peer feedback and incorporate the feedback into a final poster they present to their classmates. In the course of the project, students demonstrate the ability to clearly define a problem and how an app can help; identify appropriate input, output, storage and processing; give useful and constructive feedback to their peers; and incorporate peer feedback into their final presentation. Unit 2: Web Development Key assignment: Using the in-browser tool Web Lab, students create a multi-page web site to share their interests with their classmates. Pages are formatted using common HTML elements and CSS classes and must include text, images, stylesheets, RGB colors, links, and lists. As part of the project, students must give and receive peer feedback and incorporate it into their final product. Students must demonstrate the ability to write correct and organized code, to appropriately cite outside sources, adhere to web safety guidelines and conform to general usability heuristics. Unit 3: Programming Games and Animations: Key assignment: Using the in-browser tool Game Lab, students design and create an interactive game for their peers. Students first fill out a project guide in which they describe their game and identify the sprites, variables, and functions they will need. They then program the game using constructs such as functions, conditionals, events, and loops. After a short peer feedback process, students finalize and present their projects. Students demonstrate the ability to write readable code, use variables to store and update information, create functions to organize code, incorporate feedback, and implement the systems development process.

Digital Art:
Key Assignment: Students begin this unit with an examination of what are tessellations and frieze patterns. They practice identifying the basic constituent shapes of different patterns, and then select an appropriate pattern to recreate. After whole-class instruction that teaches students how to use the software, and individual student planning, students complete their art project 100% through digital input.

Physical Computing:
Key Assignment: Students learn in this unit the basics of electronic circuitry, and from there learn how to control what the circuit is doing (how many LEDs to light, and in which order, for example) through the application of code. Students complete two distinct phases: (1) creation of distinct electronic circuits and their controlling code in a virtual simulator, and (2): creation of these same circuits with physical components.

Future Digital World:
Key Assignment: In this unit, students explore the as-yet unrealized ideas of forward thinkers in many fields, and witness how developments in digital technology are driving these ideas. Students take extensive notes online that detail important concepts introduced in the videos and use these to drive the next phase of the project, wherein students work in groups to invent an entirely new product or service that will be useful or entertaining in the future. Finally, they develop a simple business plan that will help them to build and market their invention.

Instructional Methods and/or Strategies (REQUIRED):

Please list specific instructional methods that will be used.

- Direct Interactive Instruction
- Digital portfolio development
- Cooperative Learning Groups
- Guided Instruction
- Directed Discussion
- Socratic Seminar
- Error Analysis
- Analysis and Critique of Established Forms of Digital Art (videos, photos, apps, geometric design)
- Visual demonstrations
- Modeling of digital techniques and methods
- Critical Thinking Strategies
- Project-based Learning
- Reflective writing

Assessment Methods and/or Tools (REQUIRED):

Please list different methods of assessments that will be used.

Each unit will incorporate multiple assessment strategies. Since the class is primarily hands-on, project-based education, the majority of assessments will be project -based. Generally, each unit will have several projects, or phases. Some units will have several discrete projects, each one reflecting on a certain portion of the unit. Other units will have a scaffolded series of projects,

culminating in a major project that demonstrates the student's accumulated knowledge and abilities in each unit. Formative assessments may include quizzes, progress checks, smaller projects completed earlier in a unit, and day-to-day student teacher interaction. Final projects in each unit will be scored as summative assessments. Proficiency is demonstrated through successful completion of these projects.

COURSE PACING GUIDE AND OBJECTIVES (REQUIRED)				
Day(s)	**Objective**	**Standard(s)**	**Chapter(s)**	**Reference**
Weeks 1 - 4 (18 days)	Code.org Unit 1: Problem Solving and Computers	**CSTA K-12 Computer Science Standards (2017)** 1B-AP-08 - Compare and refine multiple algorithms for the same task and determine which is the most appropriate. 1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. 1B-AP-16 - Take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation and review stages of program development. 1B-CS-01 - Describe how internal and external parts of computing devices function to form a system. 1B-CS-02 - Model how computer hardware and software work together as a system to accomplish tasks. 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. 2-AP-17 - Systematically test and refine programs using a range of test cases. 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs. 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. Source: Code.org		
Weeks 5 - 7 (15 days)	Block Coding with Scratch and Google CS First	**ISTE Standards for Students (2016)** 1. Empowered Learner 1.a. Articulate and set personal learning goals, develop strategies leveraging technology to achieve them and reflect on the learning process itself to improve learning outcomes. 1.c. Use technology to seek feedback that informs and improves their		

		practice and to demonstrate their learning in a variety of ways. 1.d. Understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies. 3. Knowledge Constructor 3.c. Curate information from digital resources using a variety of tools and methods to create collections of artifacts that demonstrate meaningful connections or conclusions. 3.d. Build knowledge by actively exploring real-world issues and problems, developing ideas and theories and pursuing answers and solutions. 4. Innovative Designer 4.a. Know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts or solving authentic problems. 4.c. Develop, test and refine prototypes as part of a cyclical design process. 4.d. Exhibit a tolerance for ambiguity, perseverance and the capacity to work with open-ended problems. 5. Computational Thinker 5.c. Break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving. 5.d. Understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions. 6. Creative Communicator 6.b. Create original works or responsibly repurpose or remix digital resources into new creations. Source: Google CS First		
Weeks 8 - 15 (40 days)	Code.org Unit 2: Web Development	**CSTA K-12 Computer Science Standards (2017)** 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 1B-IC-18 - Discuss computing technologies that have changed the world and express how those technologies influence, and are influenced by, cultural practices. 1B-AP-11 - Decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process. 1B-AP-15 - Test and debug (identify and fix errors) a program or		

		algorithm to ensure it runs as intended. 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure. 1B-NI-05 - Discuss real-world cybersecurity problems and how personal information can be protected. 1B-AP-12 - Modify, remix or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features. 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. 3A-AP-20 - Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries. 1B-IC-21 - Use public domain or creative commons media and refrain from copying or using material created by others without permission. 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs. 2-AP-17 - Systematically test and refine programs using a range of test cases. 2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies. 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. Source: Code.org		
Weeks 16 - 18 (15 days)	Digital Visual Art	**ISTE Standards for Students (2016)** 1. Empowered Learner 1c. Use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways. 4. Innovative Designer 4a. Know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts or solving authentic problems. 4b. Students select and use digital tools to plan and manage a design process that considers design constraints and calculated risks. 4c. Develop, test and refine prototypes as part of a cyclical design process. 4d. Exhibit a tolerance for ambiguity, perseverance and the capacity to work with open-ended problems. 5. Computational Thinker 5c. Break problems into component parts, extract key information,		

		and develop descriptive models to understand complex systems or facilitate problem-solving. 5d. Understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.		
Weeks 19 - 29 (55 days)	Code.org Unit 3: Programming Games and Animation	**CSTA K-12 Computer Science Standards (2017)** 2-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies. 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 2-AP-17 - Systematically test and refine programs using a range of test cases. 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. 2-AP-12 - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs. 2-AP-18 - Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse. Source: Code.org		
Weeks 30 - 33 (20 days)	Physical Computing	**California K-12 Computer Science Standards (2018)** 6-8.CS.3 Systematically apply troubleshooting strategies to identify and resolve hardware and software problems in computing systems. 6-8.DA.9 Test and analyze the effects of changing variables while using computational models. 6-8.AP.16 Incorporate existing code, media, and libraries into original programs, and give attribution. 6-8.AP.18 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.		

Weeks 34 - 36 (15 days)	Future Digital World	**ISTE Standards for Students (2016)** 3. Knowledge Constructor 3a. Students plan and employ effective research strategies to locate information and other resources for their intellectual or creative pursuits. 3b. Students evaluate the accuracy, perspective, credibility and relevance of information, media, data or other resources. 3c. Students curate information from digital resources using a variety of tools and methods to create collections of artifacts that demonstrate meaningful connections or conclusions. 3d. Students build knowledge by actively exploring real-world issues and problems, developing ideas and theories and pursuing answers and solutions. 4. Innovative Designer 4a. Know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts or solving authentic problems. 4b. Students select and use digital tools to plan and manage a design process that considers design constraints and calculated risks. 4c. Develop, test and refine prototypes as part of a cyclical design process. 4d. Exhibit a tolerance for ambiguity, perseverance and the capacity to work with open-ended problems. 6. Creative Communicator 6c. Students communicate complex ideas clearly and effectively by creating or using a variety of digital objects such as visualizations, models or simulations. 6d. Students publish or present content that customizes the message and medium for their intended audiences.		

C. HONORS COURSES ONLY
Indicate how much this honors course is different from the standard course.

D. BACKGROUND INFORMATION
Context for course (optional)
History of Course Development (optional)