Perris Union High School District
Course of Study

A. COURSE INFORMATION

Course Title:
Computer Science Discoveries B
- ☐ 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:
Introductory Computer Science
Math, Business, Industrial Technology Ed, Computer Concepts, Application
To be completed by Human Resources only

Signature: [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	[signature]	1/9/19
Asst. Superintendent of Educational Services	[signature]	1/11/19
Governing Board		

Prerequisite(s) (REQUIRED):
None
Corequisite(s) (REQUIRED):
None
Brief Course Description (REQUIRED):
Computer Science Discoveries B takes the second 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 A (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 app development, photography and photo processing, understanding how data can be used to affect society, and synthetic music creation. Importantly, students will continue to develop their skills from CS Discoveries A with more extensive units in coding and physical computing. Students will be empowered to think of themselves confidently not only as digital consumers, but as digital creators as well. By the end of the course, students will have used fundamental computer science principles to create an app that might help solve a problem of a particular community, see the connections between data and society, extend their coding prowess, and create digital art.

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 B is designed primarily to continue development of students' skills in coding and physical computing, while concurrently allowing opportunities to solve real world problems and create fine art. The goal of Computer Science Discoveries B is to enhance students' computational thinking capacity and to increase their sense of confidence in navigating an increasingly digital world, to the point that they leave class feeling that they could contribute positively to the evolution of this world.
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 4: User Centered Design

In the User Centered Design unit, students revisit the problem solving process with an emphasis on designing products for others. Through a series of projects, students will move from thinking solely about what they would personally like in a product or app, towards discovering what users or customers other than themselves would want in a product or app. Students take on different roles in the systems development process as they move into the final project, when they design, prototype, and test an app created to meet a specific user groups needs.

Unit 5: Data and Society

In the Data and Society unit, students explore the deeply entwined relationship between computer science and data. Students learn how computers represent all information through a binary system. They begin with binary systems to represent text, images, and numbers, studying such classical representation schemes as ASCII and base two numbers, then move on to more complex representations. The unit then turns to the wider implications of data and computing as students learn how data is used to solve real world problems such as route finding and recommendation systems. The role of the Internet in the collection of data is emphasized as students look at how the collection and interpretation of crowdsourced data are changing countless industries. Throughout the process, students must work with data sets, analyzing and visualizing them both by hand and using computational tools.

Unit 6: Physical Computing

In the Physical Computing unit students continue to develop their programming skills while exploring the connections between hardware and software and the growing ubiquity of computing devices. By starting with developing event-driven apps, students will consider the various ways in which people interact with the computing devices around them. From there, students will be introduced the the Circuit Playground, a small circuit board that can be programmed in the same environment previously used for app development. Gradually students will start to integrate the various I/O elements of their circuit boards into their apps, using it as a platform to understand the difference between analog and digital data. In the culmination of this unit students will propose and prototype a unique computing device using the Circuit Playground to take input and present output.

Supplementary Units

Digital Art

Extending students' opportunities to create digital beyond geometric patterns in CS Discoveries A, this unit exposes students to digital photography and digital photo processing, as well as creating synthetic music with code. Students spend significant portions of time outside for the photography unit, taking pictures on campus based on guidelines for the day, which frequently change (for example, one day may focus on abstract shots, another day on lines and patterns, etc). Ultimately, students submit their best shots to determine the top 3 shots in each category. The synthetic music creation unit again extends exposure gained in CS Discoveries A, allowing students to strengthen their grasp of Python code while focusing on their own artistic self-expression. Students will complete scaffolded tasks that lead them to understand how sounds can be blended in nested loops of differing parameters to achieve musical effect. Finally, students compose an original piece of music solely by writing code.

Supplemental Physical Computing

In addition to the physical computing curriculum supplied by Code.org, students will enrich their learning further with supplemental projects that wed their growing knowledge of Python code with more complicated circuit diagrams and peripheral sensors. Not only will students gain an increased comprehension of hardware/software interface, they will approach the threshold of creating working robotics components from scratch.

Supplemental Coding

Students will progress from block coding and HTML tags in Computer Science Discoveries A to hybrid coding environments and ultimately to straight command line programming in Computer Science Discoveries B. Students will be presented with multiple opportunities throughout this course to develop their coding skills, not only through physical computing projects, and through the app development unit, but also by participating in tasks that focus on the "hard skills", or the vocabulary and syntax that govern the writing of good code, through online coding curriculum.

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 4: User Centered Design Key assignment: Students use App Lab to design and prototype an app to address a user's needs. They first identify the problem they will address, then research the existing apps that attempt to address that problem. Based on their research, they create a list of criteria for their app, then design and test paper prototypes with potential users. Using the test data, students then redesign and create a digital prototype to present to the class. Students demonstrate the ability to apply the systems development process, incorporate feedback, present technical information to non-technical users, and implement an event driven program. Unit 5: Data and Society Key Assignment: Students research how data is is used to impact a real world topic that they are interested in. They then define a problem within that field and describe how data could help them to solve that problem. They describe the collection, cleaning, and visualization of the data, and how the data can be used to make a decision that is clear and reasonable. They must define the algorithm they use and how it could be automated using computational tools. Throughout the process, students demonstrate the ability create meaning from data through its analysis and visualization. Unit 6: Physical Computing Key assignment: Students develop and prototype an innovation that uses the input and output elements of the Circuit Playground to enable interesting and unique forms of user interaction. Students identify a problem that they could solve using the Circuit Playground, then identify the inputs and outputs that they will need to solve the problem, as well as the relationship between them. After planning their program, students develop and code their algorithms

using the App Lab Maker Toolkit, then test the program with their peers. Students demonstrate their ability to use computers to affect the physical world with a variety of inputs and outputs, program using arrays and for-loops, and test their designs in a real world context as part of the development process.

Digital Art:
Key Assignment: Students will learn some basic guidelines for taking different types of pictures, processing them with digital tools, and selecting and storing the finalized artifacts in their portfolios. Students will be graded only on the final state of their portfolios - whether several of each type of picture are organized and displayed appropriately. This unit is one of the highlights of the year, for students and teacher alike, and several of the most gifted student-photographers will be encouraged to enter their work in various photo competitions.
Students will also be creating music synthetically by writing Python code. They will be assessed periodically on their performances on certain scaffolded tasks, and finally on their compositions of an original piece of music that meets the requirements of a rubric. Again, the most gifted of the student-composers will be invited to participate in a wider symphonic project coordinated with other schools.

Supplemental Physical Computing:
Key Assignment: Students will explore a variety of more advanced projects that underscore the interface between hardware and software; specifically, between their code and electronic circuits and sensors. Students will gain the knowledge and confidence to think about building gadgets that move and that can respond to user input.

Supplemental Coding:
Key Assignment: Overlapping with the Digital Arts unit and the Supplemental Physical Computing unit, students in this unit will continue to explore the syntax, vocabulary, and logic structures of well-written code through projects and tasks geared to that end. Students will begin to explore challenging coding tasks called "hacks" that focus on the development of pure coding skills along with their general problem-solving ability and capacity for working together with others to meet these challenges.

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 - 8 (40 days)	Code.org Unit 4: User-Centered Design	**CSTA K-12 Computer Science Standards (2017)** 2-CS-01 - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices. 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. 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-17 - Systematically test and refine programs using a range of test cases. 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs. 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. 2-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable.		

		2-DA-09 - Refine computational models based on the data they have generated. 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. 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 9 - 14 (24 days)	Code.org Unit 6: Physical Computing	**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-11 - Create clearly named variables that represent different data types and perform operations on their values. 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. 2-CS-03 - Systematically identify and fix problems with computing devices and their components. 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. 2-AP-12 - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. 2-AP-17 - Systematically test and refine programs using a range of test cases. 2-CS-01 - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices. 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. 2-CS-03 - Systematically identify and fix problems with computing devices and their components. 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse. 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		

		2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. 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-IC-21 - Discuss issues of bias and accessibility in the design of existing technologies. Source: Code.org		
Weeks 14 - 18 (20 days)	Photography	**ISTE Standards for Students (2016)** 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 6a. Students choose the appropriate platforms and tools for meeting the desired objectives of their creation or communication. 6b. Students create original works or responsibly repurpose or remix digital resources into new creations. 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.		
Weeks 19 - 24	Code.org Unit 5: Data and Society	**CSTA K-12 Computer Science Standards (2017)** 2-DA-07 - Represent data using multiple encoding schemes. 2-NI-05 - Explain how physical and digital security measures protect electronic information. 2-NI-06 - Apply multiple methods of encryption to model the secure transmission of information. 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-DA-08 - Collect data using computational tools and transform the data to make it more useful and reliable. 2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. 2-IC-23 - Describe tradeoffs between allowing information to be public and keeping information private and secure. 2-IC-22 - Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. 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. Source: Code.org		
Weeks 25 - 28 (20 days)	Supplemental Coding	**California K-12 Computer Science Standards (2018)** 6-8.DA.9 Test and analyze the effects of changing variables while using computational models. 6-8.AP.10 Use flowcharts and/or pseudocode to design and illustrate algorithms that solve complex problems. 6-8.AP.11 Create clearly named variables that store data, and perform operations on their contents. 6-8.AP.12 Design and iteratively develop programs that combine control structures and use compound conditions. 6-8.AP.13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 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 6-8.AP.19 Document programs in order to make them easier to use, read, test, and debug. 6-8.IC.22 Collaborate with many contributors when creating a computational artifact.		

Weeks 29 - 32 (20 days)	Digital Music Creation	**ISTE Standards for Students (2016)** 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 6a. Students choose the appropriate platforms and tools for meeting the desired objectives of their creation or communication. 6b. Students create original works or responsibly repurpose or remix digital resources into new creations. 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.		
Weeks 33 - 36 (20 days)	Supplemental Physical Computing	2-IC-20 - Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. 2-AP-11 - Create clearly named variables that represent different data types and perform operations on their values. 2-AP-13 - Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. 2-AP-16 - Incorporate existing code, media, and libraries into original programs, and give attribution. 2-AP-19 - Document programs in order to make them easier to follow, test, and debug. 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. 2-CS-03 - Systematically identify and fix problems with computing devices and their components. 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. 2-AP-12 - Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. 2-AP-17 - Systematically test and refine programs using a range of test cases.		

		2-CS-01 - Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices. 2-CS-02 - Design projects that combine hardware and software components to collect and exchange data. 2-CS-03 - Systematically identify and fix problems with computing devices and their components. 2-AP-14 - Create procedures with parameters to organize code and make it easier to reuse. 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 2-AP-10 - Use flowcharts and/or pseudocode to address complex problems as algorithms. 2-AP-15 - Seek and incorporate feedback from team members and users to refine a solution that meets user needs.		

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)