Date
Approved by

Course descriptions are
updated and reviewed with

Secondary Course Description

COVER PAGE

1. Course Title: Computer Science Discoveries

2. Transcript Title / Abbreviation:

3. Transcript Course Code / Number (Office Use Only):

4. School: Da Vinci Charter Academy

5. District: Davis Joint Unified School District

6. Department: Math / Elective

7. Graduation Requirement it meets:

13. Subject Area:

- □ History/Social Science
- □ English
- □ Mathematics
- □ Science
- □ Language other than English
- □ Visual & Performing Arts
- ☒ DJUSD Graduation Elective
- ☒ College Prep Elective (will seek UC/CSU approval)

8. Length of Course: 1 Year

14. Grade Level(s): 9

9. Graduation Credits:

15. UC/CSU Requirement:

10. School / District Web Site:
http://www.djusd.net
http://www.davincicharteracademy.net/

16. Seeking "Honors" Distinction? □ Yes ☒ No

11. CBEDS Course Code:

17. GPA Types:

12. School Contact : Da Vinci Charter Academy

Name: Tyler Millsap

Title/Position: Principal

Phone: (530) 757-7154 Ext.:

Fax: (530) 759-2178

E-mail: tmillsap@djusd.net

18. Credit Value:

- □ 0.5 (half year or semester equivalent)
- ☒ 1.0 (one year equivalent)
- □ 2.0 (two year equivalent)
- □ Other: ______________________________

19. Was this course previously approved by UC? □ Yes ☒ No
If so, in what year? ____________ Under what course title? ______________________________________

20. Pre-Requisites: NONE

Co-Requisites: NONE

21. Preliminary Approval - Secondary Site Principal Signature (Must be signed before proceeding to Step 22):

Tyler Millsap
____

22. Date Course Proposal with Preliminary Approval (Step 15) sent to Associate Superintendent, Educational Services: ____________

23. Review & Approval:
Date Signature
____________ Site Curriculum and Instruction Leadership Team Signature/Title ________________________________
____________ Secondary Department Articulation/Collaboration Signature/Title ________________________________

Secondary Principal Signatures: ________________________________ ________________________________

Date: ________________________________ ________________________________

BACKGROUND INFORMATION

Brief Course Description:

Computer Science Discoveries (CS Discoveries) is an introductory computer science course that empowers students to create authentic artifacts and engage with computer science as a medium for creativity, communication, problem solving, and fun. The course takes a wide lens on computer science by covering topics such as programming, physical computing, HTML/CSS, and data. The course inspires students as they build their own websites, apps, games, and physical computing devices.

Context for Course: CS Discoveries is designed to fit naturally between code.org's CS Fundamentals courses and CS Principles course. While each of these courses is designed to be an age-appropriate entry point to computer science, students with previous experience will find many new topics to explore, and they will revisit familiar topics in novel and more challenging contexts.

List the State/District Standards addressed in this course.

Currently, every lesson in CS Discoveries contains mappings to the relevant 2017 CSTA standards. The summary of all CSTA 2017 mappings can be found at **curriculum.code.org/csd/standards**

CS Discoveries was written using both the K-12 Framework for Computer Science and the draft CSTA standards as guidance. Because the revised CSTA standards have not been finalized yet, the CS Discoveries curriculum does not publish formal standards mapping documents but is built on the anticipated standards and framework. CS Discoveries will also incorporate curriculum to ensure that the course addresses any changes that may appear in the final draft.

Once the CSTA standards have been finalized and published, code.org will complete a full pass to articulate, on a unit and lesson level, its mapping to not only the CSTA standards, but also select ISTE, Common Core Math, Common Core ELA, and NGSS standards. Once this mapping has been completed, it will be available here as well as in lesson plans.

History of Course Development:

Why this Course is Relevant: Today, computing power is so ubiquitous, that all academic fields and businesses use computation methods to analyze data, solve problems, and predict future behavior. So important are these skills that "coding" is broadening well beyond computer science. Today, having coding skills will make you a better scientist, economists, analyst, manager, problem solver -- whatever you want to be. No longer a niche field, Computational Thinking (the ability to frame a problem so that it can be solved by computer resources) impacts all industries from healthcare to manufacturing to finance to art. A recent article summarizes this phenomena "Half of high-paying jobs in America now require this skill". People use software on their computers and phones everyday, if not every hour; yet, to most, coding is mystery. This course aims to remove this mystery and make computer programming a practical step in solving problems.

As articulated in the below articles, the goal of getting computer science courses in K12 education is both a national and state priority. This course will help DJUSD move in this direction.

https :// edsou rce.org/2016/gov-brown-signs-la w-to-plan-expansion-of-compu ter-science-ed ucation/569895

http://newsone.com/3545581/dems-seek-250000000-for-obama-computer-science-initiative/

COURSE GOALS AND/OR MAJOR STUDENT OUTCOMES

Description of how this course supports district goal to increase student awareness and appreciation of diversity:

Coding skills are increasingly seen a gateway to better jobs in the 21st century. Yet, many socio-economic groups are being left behind. Currently, only 18% of computer science graduates are women. Similarly, other socio-economic groups are severely underrepresented in the computer science field. While a college degree provides a good foundation for coding careers, it is not the only pathway. Code Schools, as vocational training, are rapidly growing.

One problem is that few high schools offer computer science curriculums, so many students never consider the field. Currently, DJUSD has successful coding and computer science courses that are oriented towards Robotics. This course focuses on other applications including, Web development and Data Science, to prepare students to build programs that implement creative solutions to address complex problems. It also includes the key tenets of project-based instruction including oral communication, collaboration, and developing a learning mindset. With this course, more students will learn about these skills and determine if Computer Science is something they want to pursue further either in college, at a code school, or through self-learning.

COURSE OBJECTIVES

The first semester of CS Discoveries introduces students to computer science as a vehicle for problem solving, communication, and personal expression. As a whole, this semester focuses on the visible aspects of computing and computer science, and encourages students to see where computer science exists around them and how they can engage with it as a tool for exploration and expression.

Where the first semester centers on the immediately observable and personally applicable elements of computer science, the second semester asks students to look outward and explore the impact of computer science on society. Students will see how a thorough user-centered design process produces a better application, how data is used to address problems that affect large numbers of people, and how physical computing with bare circuit boards allows computers to collect input and return output in a variety of ways.

COURSE OUTLINE

Semester 1

Unit 1 – Problem Solving: Students learn the problem-solving process, the input-output-store-process model of a computer, and how computers help humans solve problems. Students end the unit by proposing their own app to solve a problem.

Unit 2 – Web Development: Students learn to create websites using HTML and CSS inside Code.org’s Web Lab environment. Throughout the unit students consider questions of privacy, and ownership on the Internet. Students develop a personal website throughout the unit.

Unit 3 – Animations and Games: Students learn many fundamental programming constructs and practices in the JavaScript programming language while developing animations and games in Code.org’s Game Lab environment. Students end the unit by designing their own animations and games.

Semester 2

Unit 4 - The Design Process: Students apply the problem-solving process to the problems of others, learning to empathize with the needs of a user and design solutions to address those needs. During the second half of the unit students form teams to prototype an app of their own design, first on paper and eventually in Code.org’s App Lab environment.

Unit 5 - Data and Society: Students explore different systems used to represent information in a computer and the challenges and tradeoffs posed by using them. In the second half of the unit students learn how collections of data are used to solve problems and how computers help to automate the steps of this process.

Unit 6 - Physical Computing: Students use Code.org’s App Lab environment, in conjunction with the Adafruit

Circuit Playground, to explore the relationship between hardware and software. Throughout the unit, students develop prototypes that mirror existing innovative computing platforms, before ultimately designing and prototype one of their own.

TEXTS AND SUPPLEMENTAL INSTRUCTIONAL MATERIALS

Title, Author, Publisher, Edition: All curriculum resources and tutorials will forever be free to use and openly licensed under a Creative Commons license, allowing others to make derivative education resources for non-commercial purposes. [code.org] believes that every child deserves a high-quality education and that expensive curriculum should not prevent students from the opportunity to learn computer science.

Previously Adopted? □ **Yes** □ **No (If no, provide information directly below)**

Cost per book	**Total Cost**	**Budget Source**

Other:

What materials and supplies are required for CS Discoveries?

- Normal classroom supplies (pens, pencils, coloring, scissors, extra paper)
- Computers
- Worksheet Printing (There are worksheets with some lessons. Some teachers get around this by doing everything digitally.)
- Couple Decks of Cards
- Aluminum Foil
- Containers that can hold water
- Adafruit's Circuit Playground Boards and Micro USB cables. The curriculum is designed for a ratio of 2 students to 1 board & 1 usb cable.***

*** For teachers participating in our 2017-18 Professional Learning Program who are implementing the full-year version of the course, we will be subsidizing or partially subsidizing one classroom kit of Adafruit Circuit Playground Boards. This kit contains 15 boards and accessories and supports a classroom of 30 students, assuming a 2:1 ratio of students to boards. To learn more about the classroom kit, software requirements, and the details of the subsidy visit code.org/circuitplayground ***

It is important to note that the lack of expense for this course is due to the fact that Da Vinci possesses most of the infrastructure. Students already utilize school or personal laptops and wireless Internet access is provided. The primary cost will be the personnel for teaching the course. This will likely be a .2 teacher assignment. The professional learning program includes a 4 day summer workshop and 4 days throughout the school year. The professional learning program and course curriculum is provided free of charge from code.org

DIFFERENTIATED INSTRUCTIONAL METHODS AND/OR STRATEGIES

CS Discoveries is designed from the ground up to be an accessible and engaging course for all students, regardless of background or prior experience. It provides students opportunities to engage with culturally and personally relevant topics in a wide variety of contexts and aims to show all students that CS is for them.

An underlying principle belief of the course design is that acknowledging and shining a light on the historical inequities within the field of computer science is critical to reaching our goal of bringing computer science to all students. Code.org provides tools and strategies to help teachers understand and address well-known equity gaps within the field. The curriculum recognizes that some students and classrooms need more supports than others, and so those with the greatest needs should be prioritized. All students can succeed in computer science when given the right supports and opportunities, regardless of prior knowledge or privilege. The curriculum actively seeks to eliminate and discredit stereotypes that plague computer science and lead to attrition of the very students we aim to reach.

ASSESSMENT METHODS AND/OR TOOLS

Frequent assessment and feedback are critical to ensure that students are actively involved in their own learning and teachers have evidence that the class is making appropriate progress. Whether it's for adjusting the pace of the class or measuring student growth and progress, CS Discoveries has incorporated opportunities for assessment and feedback throughout the course. However, since schools have diverse grading systems, it is left up to the teacher to decide how to use the assessment resources for grading purposes.

Checks for Understanding
Throughout the lessons, teachers have many ways to quickly gauge whether students have grasped a particular concept. Almost all activities that students engage in allow for a teacher to check for understanding.

Prediction levels ask students to look at a piece of code and predict what it will do. They are often given before content is presented in order to scope students' exploration during the learning process. Teachers can monitor students' predictions and compare to their later work in the lesson.
Skill-building levels challenge students to complete a small programming task. Teachers have access to all student work in these levels and can read and run the code that students have produced. Puzzles often include exemplar solutions for teachers to reference.
Class discussions provide an opportunity for group sense making. These discussions may begin with students writing down their individual thoughts before sharing with a partner or group.
Quick-check levels include multiple choice or short answer questions. These are usually given after students have had a chance to explore a concept. They check for common misunderstandings before students move on to the next lesson or task.

Opportunities for Feedback
At the end of each lesson, teachers have at least one opportunity to assess and give feedback on what students have learned. Students may submit this work online through Code Studio or on paper.

Submittable levels are Code Studio levels that can be submitted to a teacher for feedback. Most of these levels come at the end of a lesson and involve a creation task. Teachers can use these tasks to assess whether students have mastered the objectives of the lesson and give students feedback on their progress.
Activity Guides accompany unplugged lessons in the curriculum. They include prompts and questions that teachers can use to follow students' progress through the lesson and reflection questions that can give insight into what students have learned from the activity.
Journal Questions allow students to reflect on what they have learned, and what they hope to learn more about.

End of Chapter Projects
Each chapter includes a project that incorporates the skills and understandings students have developed. These projects are designed to assess unit-specific skills and the five student practices that thread through the entire course.
Student-facing rubrics give guidance on the skills they must demonstrate, while allowing for plenty of choice in how to demonstrate what they have learned.
Flexible implementation allows teachers to incorporate different modes of presentation, such as posters and oral presentation.
Peer feedback forms prompt students to reflect explicitly on the goals of the project, as well as practice effective communication skills and iterative problem solving.

ASSESSMENT CRITERIA

The existing framework/standards for computer science will serve as the assessment criteria particularly for the first half of the course. Students will be expected to demonstrate proficiency on concepts through quizzes, tests, and activities that apply concepts. In the second semester of the course, assessment will shift towards application of concepts through real-world coding demonstrations. These will be performance-based and will be assessed using Da Vinci's Expected School-wide Learning Results.

HONORS COURSES ONLY

Indicate how this honors course is different from the standard course.