



Santa Rosa City Schools Course Proposal

Proposal Submitted By (School):

MCHS

Course Title & Course ID (Only if it is a revision or title change to a current course):

Computer Science Discoveries P

| | |
|---|---|
| In the needs statement below indicate if the course is a: | Answer Below: |
| Addition, Revision, Deletion, Pilot, or Title Change? (Pick one) | Addition (Replace Intro to Computer Science Principles) |
| What year will the course be initially offered? | 2020-2021 |
| What prerequisite, if any, are there for this course and how does the course fit into continuous improvement at your school site? | None |

Needs Statement: Discuss how this course fits into your Site and/or the District's goals. Attach minutes of meetings where this course was approved.

Maria Carrillo High School has been offering a computer science course for the past three years. We had used the course title of Computer Foundations. When reviewing the course description, the course we had offered did not fit the Computer Foundations description and is more of a computer science curriculum. A student survey was conducted at MCHS showing 238 students interested in taking a computer science course. We would like to offer an entry level computer science course. In the 2020-2021 school year, if student interest continues, we would like to build a course sequence of **Computer Science Discoveries**, **Computer Science Principles** (which is an honors course through Code.org) then potentially **AP Computer Science Principles** for the 2021-2022 school year.

Graduation Requirements: Specify which requirement is met. (High School only)

Elective

UC a-g Requirements: Specify which requirement is met. (High School only)

G Elective

Explain the rationale for course addition or modification

We want to provide a course to meet the needs of our students and promote STEAM skills necessary for students to be successful in the high-tech industry. We would like to use the course title **Computer Science Discoveries**.

Explain the measurable learning outcomes

Common Core Math Standards

- **5.OA.1** – Use parentheses, brackets, or braces in numerical expressions, and evaluate expressions with these symbols.
- **5.OA.2** – Write simple expressions that record calculations with numbers, and interpret numerical expressions without evaluating them. For example, express the calculation “add 8 and 7, then multiply by 2” as $2 \times (8 + 7)$. Recognize that $3 \times (18932 + 921)$ is three times as large as $18932 + 921$, without having to calculate the indicated sum or product.
- **7.G.1** – Solve problems involving scale drawings of geometric figures, including computing actual lengths and areas from a scale drawing and reproducing a scale drawing at a different scale.
- **8.F.1** – Understand that a function is a rule that assigns to each input exactly one output. The graph of a function is the set of ordered pairs consisting of an input and the corresponding output.¹
- **8.F.2** – Compare properties of two functions each represented in a different way (algebraically, graphically, numerically in tables, or by verbal descriptions). For example, given a linear function represented by a table of values and a linear function represented by an algebraic expression, determine which function has the greater rate of change.
- **A.SSE.1** – Interpret expressions that represent a quantity in terms of its context.
- **A.SSE.2** – Use the structure of an expression to identify ways to rewrite it. For example, see $x^4 - y^4$ as $(x^2)^2 - (y^2)^2$, thus recognizing it as a difference of squares that can be factored as $(x^2 - y^2)(x^2 + y^2)$.
- **A.CED.1** – Create equations and inequalities in one variable and use them to solve problems. Include equations arising from linear and quadratic functions, and simple rational and exponential functions.
- **A.CED.2** – Create equations in two or more variables to represent relationships between quantities; graph equations on coordinate axes with labels and scales.
- **F.IF.1** – Understand that a function from one set (called the domain) to another set (called the range) assigns to each element of the domain exactly one element of the range. If f is a function and x is an element of its domain, then $f(x)$ denotes the output of f corresponding to the input x . The graph of f is the graph of the equation $y = f(x)$.
- **F.IF.2** – Use function notation, evaluate functions for inputs in their domains, and interpret statements that use function notation in terms of a context.
- **F.IF.3** – Recognize that sequences are functions, sometimes defined recursively, whose domain is a subset of the integers. For example, the Fibonacci sequence is defined recursively by $f(0) = f(1) = 1$, $f(n+1) = f(n) + f(n-1)$ for $n \geq 1$.

Common Core Math Practices

- **MP.1** – Make sense of problems and persevere in solving them.
- **MP.2** – Reason abstractly and quantitatively.
- **MP.3** – Construct viable arguments and critique the reasoning of others.
- **MP.4** – Model with mathematics.
- **MP.5** – Use appropriate tools strategically.
- **MP.6** – Attend to precision.
- **MP.7** – Look for and make use of structure.
- **MP.8** – Look for and express regularity in repeated reasoning.

Course Description (To be used in the course catalog)

Computer Science Discoveries is a full-year survey course that takes a wide lens on computer science by covering topics such as programming, physical computing, HTML/CSS, user interface design, and data. Students are empowered to create authentic artifacts and engage with CS as a medium for creativity, communication, problem solving, and fun. By the end of the course, students will have used the software development process and fundamental programming constructs to design apps, create web pages, develop games, use data to solve problems, and program interactions with the physical world through a variety of sensors and hardware. Throughout the course, students are encouraged to reflect on what they have learned about computer science and how it affects their world through journal prompts, classroom presentations, and written descriptions of digital and physical artifacts that they create.

Detailed Course Design

(Course design should include the objectives, activities, assessments, and standards to be addressed in this course.)

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 Assignment(s):

Students identify and define a real world problem and brainstorm ways an app could be used to help solve the problem. They then complete a multi-page report identifying the inputs, outputs, storage and processing needed, as well as create a sketch of the interface. Students then write up peer feedback reports and incorporate this 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. They must use appropriate presentational writing and speaking in presenting their ideas.

Unit Lab Activities:

Students work in groups to design aluminum foil boats that will support as many pennies as possible. Groups have two rounds to work on their boats, with the goal of trying to hold more pennies in the second than they did in the first. Afterwards, students reflect on the strategies that they used, and how those strategies were refined over the course of the experience. Finally, students formalize their strategies into a general problem solving process that can be used for a variety of problems. They produce a one to two page write up that explains the role of each part of their problem solving process and the part it played in completing the activity

successfully.

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 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 Assignment(s):

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 Lab Activities:

Using the inspector tool, students investigate the relationship between HTML code and the web pages that it produces. Students explore working code, then modify the code to fit given criteria as well as their personal preferences. They then reflect on the value of using a markup language to separate out the content and structure of a digital artifact and document what they have learned about the particular tags and syntax of the language. Last, they use what they have learned to create their own web pages.

Unit 3: Interactive 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, using the concepts to model movement in the real world. 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.

Unit Assignment(s):

Using the in-browser tool Game Lab, students design and create an interactive game for their peers. Students first complete 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.

Unit Lab Activities:

Students investigate the relationships between acceleration, velocity and position and how to use their knowledge to model natural phenomena such as gravity and parabolic movement. Inside the Game Lab environment, students predict how a constant change in velocity over time will impact the movement of a sprite, observing acceleration in the same and opposite direction of movement. Students then program their own types of movement, using what they have learned to model gravity and jumping, with movement in one and two dimensions. Last, students write up the various types of movements they were able to model and what properties they manipulated in order to do so, reflecting on how they were able to use the concepts of velocity and acceleration to model complex movement in two dimensions.

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, and 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 Assignment(s):

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 write up the 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. They also must communicate clearly about their app and using both interpersonal and presentational language.

Unit Lab Activities:

Students design an app interface to meet a user's needs and create paper prototype in order to meet that need. After conducting user interviews and collecting analyzing the results, students determine the minimum set of features their app would need in order to address their user's needs and create a paper prototype that includes those features. They then test their interface by observing their user interact with the prototype to complete several simulated tasks. Afterwards, students write up the results of their tests and determine what changes they should make in the design, proposing a improved interface for their app.

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 Assignment(s):

Students design a recommendation engine based on data that they collect and analyze from their classmates. After looking at an example of a recommendation app, students complete a write up for a potential automated recommendation system and the data that it would use. They then create a survey, collect information about their classmates' choices, interpret the

data, and use what they have learned to create a recommendation algorithm. After testing their algorithm on potential users, students complete a written reflection on the benefits and drawbacks of automated decision making and how it could be improved. They then make any necessary updates to their projects before preparing a presentation to the class.

Unit Lab Activities:

Students design a method to store information using a fixed set of elements. They work in small groups to create a system that can represent any letter in the alphabet using only a single stack of cards. Each card has one of six different possible drawings and so to represent the entire alphabet students will need to use patterns of multiple cards to represent each letter. Students create messages with their systems and exchange with other groups to ensure the system worked as intended. In the wrap-up reflection the students review any pros and cons of the different systems. They discuss commonalities between working systems and determine the characteristics needed for a good system of information representation.

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 to 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.

Unit Assignment(s):

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 complete a report that describes a problem that they could solve using the Circuit Playground and identifies the inputs and outputs that they will need to solve the problem, as well as the relationship between them. After planning their programs, 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.

Unit Lab Activities:

Students design and program a game that uses the elements of a physical computing device. In small groups, students brainstorm and design a small game that can be played using the input and output elements of the Circuit Playground. They then determine the components of the board that they will need to use and how their program will interact with those components, including the events and functions that their software will need. They then develop the game and share it with their peers, collecting feedback on its design. Last, they analyze the feedback and determine what improvements should be made for the next iteration of the software.

Budget

| Projected Costs | Start-up | Ongoing |
|--|------------|---|
| Personnel (Not to include classroom instructor unless a new section is needed) | | Section dependent upon student course requests. |
| Instructional Material Supplies per student (textbooks, software, etc.) | | Using chromebooks |
| Services (training, equipment maintenance, contracts, etc.) | | N/A |
| Capital Outlay (remodeling, technology, etc.) | | N/A |
| Total Projected Costs | \$0 | \$0 |

Instructional Materials

| Type | Publisher | Title | ISBN | Author | Copyright | # Have/Need |
|---------|-----------|------------------------------|------|----------|-----------|-----------------|
| Website | Code.org | Computer Science Discoveries | | Code.org | | Free Curriculum |
| | | | | | | |
| | | | | | | |
| | | | | | | |

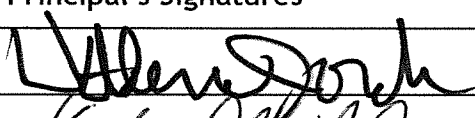
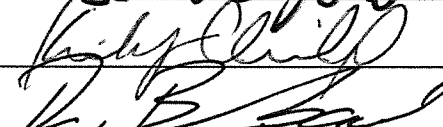

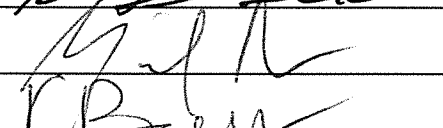
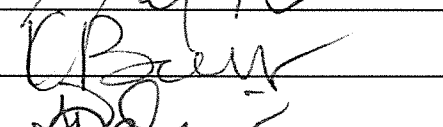

Funding Source(s) for Costs and Instructional Materials

| | |
|---|-----|
| Grants (indicate specific grant and grant timeline) | N/A |
| Categorical Funds (include related programs) | N/A |
| Career Technical Education (must be for an approved CTE course) | N/A |
| Department Funds | N/A |
| Other (be specific) | |

Appendix of Additional Documents

| |
|--|
| <u>* Required additional documents include meeting minutes where the course was discussed and approved</u> |
| AC Minutes November 6, 2019 Code.org Computer Science Discoveries Curriculum |

District Principal Review and Approvals:

| Principal's Signatures | Site | Approved / Not Approved |
|---|------|-------------------------|
|  | RHS | Approved |
|  | SRHS | Approved |
|  | MHS | Approved |
|  | EAHS | approved |
|  | MCIS | approved |
|  | PHS | approved |

District Department Chair Review and Approvals:

| Department Chair Signatures | Site | Approved / Not Approved |
|-----------------------------|------|-------------------------|
| | | |
| | | |
| | | |

| | | |
|--|--|--|
| | | |
| | | |
| | | |