SANTA ROSA CITY SCHOOLS
Excellence is our Common Ground

Santa Rosa City Schools Course Proposal

Course: Computer Science Principles HP

Proposal Submitted By Maria Carrillo High School: Teaching and Learning

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. For the 2019-2020 school year we offered **Introduction to Computer Science Principles**. In the 2020-2021 school year, if student interest continues, we would like to build a course sequence of **Computer Science Discoveries**, **Computer Science Principle**s (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)

This course will meet the elective science requirement for graduation.

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

This course will be submitted to the UC to meet D requirement in the A-G requirements.

Explain the rationale for course addition or modification

We are modifying the Computer Foundations 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 Principles HP.**

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 × (8 + 7). Recognize that 3 × (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 x4 – y4 as (x2)2 – (y2)2, thus recognizing it as a difference of squares that can be factored as (x2 – y2)(x2 + y2).
- **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 ≥ 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.

Computer Science Standards

Course Description (To be used in the course catalog)

Computer Science Principles introduces students to the foundational concepts of computer science and challenges them to explore how computing and technology can impact the world. More than a traditional introduction to programming, it is a rigorous, engaging, and approachable course that explores many of the foundational ideas of computing so all students understand how these concepts are transforming the world we live in. Topics include the internet, digital information, programming, big data, cryptography, app development, and the global impacts of computing. This course is identical to the AP Computer Science Principles curriculum offered by Code.org, but for students who opt not to take the AP exam.

Detailed Course Design

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

Internet

In this unit students learn how computers represent all kinds of information and how the Internet allows that information to be shared with millions of people.

The first chapter (Lessons 1-7) explores the challenges and questions that arise when representing information in a computer or sending it from one computer to another. It begins by investigating why on-off signals, also known as binary signals, are used to represent information in a computer. It then introduces the way common information types like text and numbers are represented using these binary signals. Finally, it illustrates the importance of establishing shared communication rules, or protocols, for successfully sending and receiving information.

In the second chapter (Lessons 8-14), students learn how the design of the internet allows information to be shared across billions of people and devices. Making frequent use of the Internet Simulator, an interactive simulator of different internet protocols, they explore the problems the original designers of the internet had to solve and invent their own solutions. To conclude the unit, students research a modern social dilemma driven by the ubiquity of the internet and the way it works.

Unit Assignments:

Students will research and prepare a flash talk about a social issue related to the Internet. Students pick one of: Net Neutrality, Internet Censorship, or Computer/Network Surveillance. Students will then conduct online research about their topic. Throughout this process students must read and determine the quality and value of the digital sources they find. Students will then write a two minute, or roughly 300 words, presentation that explains technical details of their topic, explains its broader social impact, and offers an opinion on how dilemmas arising from their topic should be handled. Students will submit a written version of their flash talk along with a completed research guide citing their sources and the information used from each. They will then present their flash talk to their peers. Students will be assessed on the clarity of their writing, their appropriate use of concepts covered in the unit, the strength of their arguments, and the demonstrated thoroughness of their research.

Lab Activity:

In Lesson 10: Routers and Redundancy, students use a version of the Internet Simulator that enables them to send messages back and forth across a simulated network and subsequently investigate a log of how the traffic was routed. With the support of their teacher, students analyze data captured in router logs to draw conclusions about how traffic on the Internet is routed dynamically across many computers before arriving at its ultimate destination. Using a provided activity guide they then compose written responses reflecting on the benefits and potential privacy concerns that arise from routing traffic in this way.

Digital Information

This unit explores the way large and complex pieces of digital information are stored in computers and the associated challenges. Through a mix of online research and interactive widgets, students learn about foundational topics like the size of digital files, the way these files are compressed, image representation, metadata, the difference between lossy and lossless compression, and the advantages and disadvantages of different file formats. At the conclusion of the unit students will research a real world file format and create a presentation arguing why this format's features make it better than other competing formats.

Assignments:

In the final lesson of this unit, "Rapid Research - Format Showdown", students will research a popular file format either currently in use or from history. Students must read and determine the quality and value of digital sources that they find. They will then synthesize their research to complete a "one-pager" research guide (~300 words) that summarizes their findings. They will also design a digital artifact (video, audio, graphic, etc.) that succinctly summarizes the advantages of their format over other similar ones. Students will practice research skills, synthesizing information, and presenting information in a succinct and approachable way. Students will be assessed on the thoroughness of their research, the strength of the argument presented by their digital artifact, and the clarity and correctness of the information presented in their one-pager.

Lab Activity:

In Lesson 2: Text Compression, students explore how lossless text compression reduces the size of a file without losing any information. Students use a Text Compression Widget that allows them to manually compress short pieces of text and observe how different approaches allow them to achieve different rates of compression. This activity highlights not only the mechanics underlying many modern text compression algorithms, but also need for heuristic approaches to computationally hard problems where "good enough" solutions are deemed acceptable. At the conclusion of the lesson students write responses to a set of reflection prompts that highlight these core concepts.

Intro to Programming

In this unit students explore the fundamental topics of programming, algorithms, and abstraction as they learn to programmatically draw pictures in App Lab. An unplugged sequence at the beginning of the unit highlights the need for programming languages as well as the creativity involved in designing algorithms. Students then begin working in App Lab where they use simple commands to draw shapes and images using a virtual "turtle." As they're introduced to more complex commands and programming constructs, students learn to break down programming problems into manageable chunks. The unit ends with a collaborative programming project to design a digital scene.

Unit Assignment(s):

In the final lesson of the unit, "Practice PT - Design a Digital Scene", students will design a program that draws a digital scene of their choosing. Students will be working in groups of 3 or 4 and will begin by identifying a scene they wish to create. They will then use Top-Down Design to identify the high-level functions necessary to create that image. The group will then assign these components to individual members of the group to program. After programming their individual portion, students will combine all of their code to compose the whole scene. The project concludes with reflection questions similar to those students will see on the Create Performance Task. Students will learn to apply the programming constructs and practices they've learned throughout the unit, as well as how to collaborate in their various roles as part of the systems development process. Students will individually write responses to three questions (~450 words) in which they identify the purpose of their program, describe the systems development process used to create it, and identify a programming abstraction they created in its development. Students are assessed both on the appropriate use of programming constructs in their program code and the mastery of programming concepts and practices demonstrated in their written submission.

Unit Lab Activities:

In Lesson 8: Creating Functions with Parameters and Lesson 9: Looping and Random Numbers, students design programs that use procedural abstraction and looping to generate complex and expressive programs. Students explore the complementary ways these two tools allow programmers to repeat and encapsulate code as the work to develop complex drawings using the App Lab programming environment. Students then analyze a complex drawing to identify components and patterns that correspond to the programming constructs that they have available to them, and use those constructs to programmatically generate the drawing. At the conclusion of the activity students submit their final complex drawing as well as a set of written reflection prompts explaining the value of these two programming constructs and the different contexts in which it makes most sense to use each.

Big Data and Privacy

In this unit students explore the technical, legal, and ethical questions that arise from computers enabling the collection and analysis of enormous amounts of data. In the first half of the unit, students learn about both the technological innovations enabled by data and the privacy and security concerns that arise from collecting it. In the second half of the unit, students learn how cryptography can be used to help protect private information in the digital age. Students will explore these topics through a combination of research and exploring interactive digital tools, or "widgets". This unit features two significant research projects.

Unit Assignment(s):

In Lesson 10, "Rapid Research - Cybercrime", students learn about various types of cybercrimes and the cybersecurity measures that can help prevent them. Then students conduct online research to investigate a cybercrime event of their choosing with a particular focus on the data that was lost or stolen and the concerns that arise as a result. Students must read and determine the quality and value of digital sources they find online. They will then synthesize their research to complete a "one-pager" research guide (~300 words) that summarizes their findings. They will also design a digital artifact (video, audio, graphic, etc.) that summarizes the cybercrime event itself, the type of attack used, the way it could've been prevented, the data that was leaked, and the privacy or security concerns that arise. Students will practice research skills, synthesizing information, and presenting information in a succinct and approachable way. Students will be assessed on the thoroughness of their research and the clarity and correctness of the information presented in their one-pager.

Unit Lab Activities:

In Lesson 2: Finding Trends with Visualizations, students explore time-series search data collected through the Google Trends tool in order to identify patterns and relationships in the popularity of different web search terms. Students begin with hypotheses of potential patterns or interesting relationships that might exist within the data. Students then practice exploring large data sets and refining data visualizations in order to uncover meaningful patterns. Finally, they respond to a series of written prompts that ask them to describe the data patterns they've found and interpret them as meaningful conclusions or stories.

Building Apps

This unit continues to develop students' ability to program in the JavaScript language using App Lab. Students create a series of simple applications (apps) that live on the web, each highlighting a core concept of programming. In the first half of the unit students learn to design apps that respond to user interaction like clicks and key presses. Concepts introduced in this chapter include variables, user input, text strings, Boolean expressions, and if-statements.

In the second half of the unit students learn to program with larger and more complex data structures. Early in the chapter students return to the study of loops, this time using them to simulate real world events. Next they learn to program with lists of information in order to develop apps that store and process large amounts of data. They will also learn to compare the efficiency of different list-processing algorithms. The final lessons of the chapter introduce advanced programming topics including return values and objects. The chapter concludes with a self-directed project in which students apply all the programming skills and concepts they've learned in the course.

Unit Assignment(s):

Lesson 10, "Building an App - Color Sleuth", asks students to walk through the iterative development of building an app from scratch. Following an imaginary conversation between two characters - Alexis and Michael - students follow the problem solving and program design decisions they make for each step of constructing the app. Along the way they decide when and how to break their program into functions, how to design the logic of the game, and what extensions to the core project they would like to add.

Once students complete their program they will must answer a series of free response questions (~300 words) in which they explain how the concepts of abstraction and algorithms apply to the project they completed.

Students' work will be assessed on the appropriate use of programming concepts as well as their mastery of foundational programming concepts as demonstrated in their written submissions.

This project is one of several similar "Building an App" projects found in this unit.

Unit Lab Activities:

In Lesson 12: Loops and Simulations, students design a program that will simulate the flipping of thousands of coins in order to collect data about the patterns that arise. Students begin by developing hypotheses about how many flips it will take to get 10,000 total heads and how many flips it will take to get a streak of twelve heads in a row. Students then model flipping coins through the design and creation of a simulation program to test their hypotheses and compare them to the results of their simulation. At the conclusion of the lesson they must extrapolate from the data collected to predict how many flips they believe it would take to get 10,000,000 total heads or a streak of 20 flips, supporting their conclusions with the analysis of data that their simulations generated.

Data Tools

In this unit students develop skills interpreting visual data and using spreadsheets and visualization tools to create their own data visualization and accompanying summary. Through an ongoing project - the "class data tracker" - students learn how to collect and clean data, and to use common techniques for computing aggregations and creating visualizations. Throughout the unit students practice critiquing conclusions drawn from data and learn to identify assumptions underlying the analysis of a particular data set or data visualization. To conclude the unit students analyze the data they've collected about themselves to design a data visualization and explain what the visualization demonstrates.

Unit Assignment(s):

In Lesson 7, "Practice PT - Tell a Data Story" students will analyze the data that they have been collecting as a class in order to demonstrate their ability to discover, visualize, and present a trend or pattern they find in the data. Leading up to this lesson, students will have been working in pairs to clean and summarize data collected by their class using the spreadsheet tools and techniques they've practiced throughout the unit. In this project they will explore those summaries to find meaningful trends and then create a data visualization that highlights that story. They will write a summary of their findings (~500 words) that describes the process used to analyze their data, the trends they have found as a result, a possible explanation for this trend, any assumptions impacting their analysis, and a recommendation they could make based on their analysis. This project will then be presented to the class. Students will learn all the steps necessary to turn raw data into useful conclusions, as well as the skills necessary to communicate about those conclusions to others. They will be assessed on the quality of the data visualization itself as well as the clarity, correctness, and mastery of key data analysis concepts and skills demonstrated in their written submission.

Unit Lab Activities:

In Lesson 4: Discover a Data Story, students are given a number of cleaned data sets and use data visualization tools to identify meaningful patterns and relationships. Students use analysis and graphing tools provided in most spreadsheet software (e.g. Excel or Sheets) to explore these data sets and then use a provided activity guide to present their findings and reflect on the potential meaning of the patterns they find. This activity supports them in building skills and understandings related to data analysis that they will use when analyzing their class data collected as part of the Practice PT they will complete in Lesson 7.

Honors Final Exam Details:

For a first semester final project students will complete the Explore Performance Task. In this task, students will identify a computing innovation, research the innovation, identify the quality and value of the digital sources they find, synthesize information about the innovation's key features, write roughly 700 words explaining the conclusions of their research, and then design a digital artifact (slide, data visualization, video, or audio recording) that helps further explain the functionality of the innovation and its purpose. Throughout this project students will pay particular attention to the purpose of the innovation, the beneficial and harmful effects it has on society and the economy, and any privacy, security, or data storage concerns that arise from the innovation. Students complete this task over the course of 8 class hours which they can allocate as they see fit. Students are assessed both on the clarity, quality, and effectiveness of their digital artifact as well as the mastery of key computer science concepts demonstrated in their written submissions.

To conclude the course students complete the Create Performance Task. In this task, students will independently propose, design, program, and complete a programming project. They will then compose a roughly 750 word write-up describing the purpose of the program, the way it accomplishes that purpose, the process used to develop the program, and the way key programming concepts and constructs were incorporated into its design. Students complete this task over the course of 12 class hours which they can allocate as they see fit. Students are assessed on their mastery of key programming concepts, as demonstrated in their program code, as well as their ability to correctly apply and describe programming processes and constructs as demonstrated in their written submission.

The final exam is a two hour comprehensive exam that includes 74 single select and multiple select multiple choice questions. Multiple-choice questions on the exam are classified according to learning objectives within each big idea in the Computer Science Principles curriculum framework. Some exam questions may be aligned to more than one learning objective. For example, a question on programming might implement an algorithm and contain abstractions. This exam is parallels the content and structure of the AP CS Principles exam, and also serves as a mock/practice exam for students who choose to take the AP CS Principles exam.

Budget

Projected Costs	Start-up	Ongoing
Personnel (Not to include classroom instructor unless a new section is needed)		Dependent upon course requests
Instructional Material Supplies per student (textbooks, software, etc.)		Using chromebooks and raspberry systems
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
Code.org	CODE.ORG	Computer Science Principles			2018	Internet based - have what we need

Already own instructional materials.

Funding Source(s) for Costs and Instructional Materials

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

Appendix of Additional Documents

AC Meeting Minutes November 6, 2019
Code.org Computer Science Curriculum Guide

District Principal Review and Approvals:

Principal's Signatures	Site	Approved / Not Approved
[illegible]	PHS	approved
[illegible]	RHS	Approved
[illegible]	SRHS	Approved
[illegible]	MHS	Approved
[illegible]	EAHS	Approved
[illegible]	MCHS	Approved

District Department Chair Review and Approvals:

Department Chair Signatures	Site	Approved / Not Approved
[illegible]	SRHS	approved
[illegible]	PHS	APPROVED.
Doug Benenson	MHS	Approved
[illegible]	RHS	Approved
[illegible]	EAHS	Approved
[illegible]	MCHS	Approved