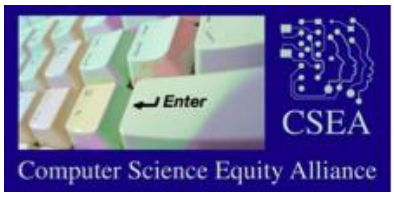
Joanna Goode
University of Oregon

Gail Chapman
University of California, Los Angeles

Sponsors & Supporters

This curriculum was created under the auspices of the Broadening the Participation in Computing National Science Foundation grant, "Into the Loop: An University K-12 Alliance to Increase and Enhance the Computer Science Learning Opportunities for African-American, Latino/a, and Female Students in the Second Largest School District in the Country". Principal Investigator: Jane Margolis (UCLA); Co-Principal Investigators Joanna Goode (University of Oregon), Todd Ullah (LAUSD), Deborah Estrin (UCLA). The Computing and Data Analysis Unit was created under the auspices of the National Science Foundation Math/Science Partnership grant, "MOBILIZE: Mobilizing for Innovative Computer Science Teaching and Learning." Co-principal Investigators: Deborah Estrin (UCLA, CENS), Mark Hansen (UCLA, CENS), Joanna Goode (University of Oregon, College of Education), Jane Margolis (UCLA, Center X), Thomas Philip (UCLA, Center X), Jody Priselac (UCLA, Center X), and Todd Ullah (LAUSD).



Acknowledgments

George Benainous, David Bernier, Robb Cutler, Judy Hromcik, Michelle Hutton, John Landa, Clifford Lee, Cueponcaxochitl Moreno, Jean Ryoo, Suzanne Schaefer, Chris Stephenson, Diane Watkins

For additional information related to the Exploring Computer Science Partnership visit: www.exploringcs.org

CONTENTS

Course Overview

Goals

Exploring Computer Science is designed to introduce students to the breadth of the field of computer science through an exploration of engaging and accessible topics. Rather than focusing the entire course on learning particular software tools or programming languages, the course is designed to focus the conceptual ideas of computing and help students understand why certain tools or languages might be utilized to solve particular problems. The goal of *Exploring Computer Science* is to develop in students the computational thinking practices of algorithm development, problem solving and programming within the context of problems that are relevant to the lives of today's students. Students will also be introduced to topics such as interface design, limits of computers and societal and ethical issues.

This curriculum has been developed for a culturally, linguistically, and socially diverse group of students in Los Angeles Unified School District. District-wide, student ethnicities include .3% American Indian, 3.7% Asian, .4% Pacific Islander, 2.3% Filipino, 73.0% Latino, 10.9% African American, 8.8% White, and .6% Other or multiple responses. Over 38% of students are English-language learners, with most English language learners' students speaking Spanish as their primary language. Furthermore, 74% of students qualify for free or reduced lunches.

Standards

The standards used for the *Exploring Computer Science* curriculum are based on the topics and goals outlined in *A Model Curriculum for K-12 Computer Science* (2006) developed by the ACM K-12 task force curriculum committee. Most of the objectives in the course align with the Level III course, *Computer Science as Analysis and Design*, while some objectives are necessarily aligned with the Level II course, *Computer Science in the Modern World*, in order to provide appropriate background knowledge for the more advanced topics.

Hardware

An ideal laboratory environment for this course would include one computer for each student in the class. These computers can be either Macintosh or PC depending on availability. A networked system would make installation of software easier for the teacher.

Software

Each computer in the classroom should have a web browser installed that allows students to perform searches and make use of a variety of websites and internet tools. Teachers will need to download and install the Scratch programming language available at http://www.scratch.mit.edu. All website url's were updated at the time of this writing, but may change over time.

Prerequisites

This course will be considered a college preparatory elective for California students. It is recommended that students have completed an Algebra course prior to enrolling. Thus, the course should provide a rigorous, but accessible, introduction to computer science. No previous computer science course is required to take this course.

The Instructional Philosophy of *Exploring Computer Science*

Introduction to Curricular Approach

Exploring Computer Science teaches the creative, collaborative, interdisciplinary, and problem-solving nature of computing with instructional materials which feature an inquiry-based approach to learning and teaching. As part of this curriculum, students will delve into real-world computing problems that are culturally-relevant and address social and ethical issues while delivering foundational computer science knowledge to students. Students will engage in several in-depth projects to demonstrate the real-world applications of computing.

This curriculum builds off of learning theories that view learning as a social and cultural process that does not only occur in a vacuum at school; that is, students bring to school bodies of knowledge from their lives, culture, and communities. Building from students' prior knowledge, the collection of problem solving skills, everyday "algorithmic thinking", and social and ethical knowledge of computer-related problems will result in a more student-centered curriculum. Each unit connects students' informal knowledge, technology skills, and beliefs about computing to the theoretical and foundational tenets of computer science. Students will become members of a "computing community of practice" in the classroom where they will be introduced to the behavior, language, and skills of computer scientists. Furthermore, the interdisciplinary nature of computing allows for the incorporation of subject-matter topics across disciplines into the computing curriculum.

The Nine Principles of Learning from the Institute for Learning provide the theoretical foundation of research-based instructional practices that provide the foundation for the Secondary Redesign Comprehensive Plan developed by the Los Angeles Unified School District. These nine principles underscore the beliefs of the Los Angeles Unified School District; they are integrated throughout and explain the pedagogy used in the course.

1. Organizing for Effort

An effort-based school replaces the assumption that aptitude determines what and how much students learn with the assumption that sustained and directed effort can yield high achievement for all students. Everything is organized to evoke and support this effort, to send the message that effort is expected and that tough problems yield to sustained work. High minimum standards are set and assessments are geared to the standards. All students are taught a rigorous curriculum aligned to the standards, along with as much time and expert instruction as they need to meet or exceed expectations. This principle is one of the guiding beliefs common in every school in the Los Angeles Unified School District.

2. Clear Expectations

If we expect all students to achieve at high levels, then we need to define explicitly what we expect students to learn. These expectations need to be communicated clearly in ways that get them "into the heads" of school professionals, parents, school communities and, above all, students themselves. Descriptive criteria and models of work that meets standards should be publicly displayed, and students should refer to these displays to help them analyze and discuss their work. With visible accomplishment

targets to aim toward at each stage of learning, students can participate in evaluating their own work and setting goals for their own efforts.

3. Fair and Credible Evaluations

If we expect students to put forth sustained effort over time, we need to use assessments that students find fair, and that parents, community, and employers find credible. Fair evaluations are ones that students can prepare for: therefore, tests, exams and classroom assessments as well as the curriculum must be aligned to the standards. Fair assessment also means grading against absolute standards rather than on a curve, so students clearly see the results of their learning efforts. Assessments that meet these criteria provide parents, colleges, and employers with credible evaluations of what individual students know and can do.

4. Recognition of Accomplishment

If we expect students to put forth and sustain high levels of effort, we need to motivate them by regularly recognizing their accomplishments. Clear recognition of authentic accomplishment is the hallmark of an effort-based school. This recognition can take the form of celebrations of work that meets standards or intermediate progress benchmarks en route to the standards. Progress points should be articulated so that, regardless of entering performance level, every student can meet real accomplishment criteria often enough to be recognized frequently. Recognition of accomplishment can be tied to an opportunity to participate in events that matter to students and their families. Student accomplishment is also recognized when student performance on standards-based assessments is related to opportunities at work and in higher education.

5. Academic Rigor in a Thinking Curriculum

Thinking and problem solving will be the "new basics" of the 21st century, but the common idea that we can teach thinking without a solid foundation of knowledge must be abandoned, so must the idea that we can teach knowledge without engaging students in thinking. Knowledge and thinking are intimately joined. This implies a curriculum organized around major concepts that students are expected to know deeply. Teaching must engage students in active reasoning about these concepts. In every subject, at every grade level, instruction and learning must include commitment to a knowledge core, high thinking demand, and active use of knowledge.

6. Accountable Talk

Talking with others about ideas and work is fundamental to learning but not all talk sustains learning. For classroom talk to promote learning it must be accountable to the learning community, to accurate and appropriate knowledge, and to rigorous thinking. Accountable talk seriously responds to and further develops what others in the group have said. It puts forth and demands knowledge that is accurate and relevant to the issue under discussion. Accountable talk uses evidence appropriate to the discipline (e.g., proofs in mathematics, data from investigations in science, textual details in literature, and documentary sources in history) and follows established norms of good reasoning. Teachers should intentionally create the norms and skills of accountable talk in their classrooms.

7. Socializing Intelligence

Intelligence is much more than an innate ability to think quickly and stockpile bits of knowledge. Intelligence is a set of problem solving and reasoning capabilities along with the habits of mind that lead one to use those capabilities regularly. Intelligence is equally a set of beliefs about one's right and obligation to understand and make sense of the world, and one's capacity to figure things out over time. Intelligent habits of mind are learned through the daily expectations placed on the learner by calling on students to use the skills of intelligent thinking, and by holding them responsible for doing so, educators can "teach" intelligence. This is what teachers normally do with students from whom they expect achievement; it should be standard practice with all students.

8. Self-management of Learning

If students are going to be responsible for the quality of their thinking and learning, they need to develop and regularly use an array of self-monitoring and self-management strategies. These meta- cognitive skills include noticing when one doesn't understand something and taking steps to remedy the situation, as well as formulating questions and inquiries that let one explore deep levels of meaning. Students also manage their own learning by evaluating the feedback they get from others; bringing their background knowledge to bear on new learning; anticipating learning difficulties and apportioning their time accordingly and judging their progress toward a learning goal. These are strategies that good learners use spontaneously and that all students can learn through appropriate instruction and socialization. Learning environments should be designed to model and encourage the regular use of self-management strategies.

9. Learning as Apprenticeship

For many centuries most people learned by working alongside an expert who modeled skilled practice and guided novices as they created authentic products or performances for interested and critical audiences. This kind of apprenticeship allowed learners to acquire complex interdisciplinary knowledge, practical abilities, and appropriate forms of social behavior. Much of the power of apprenticeship learning can be brought Into schooling by organizing learning environments so that complex thinking is modeled and analyzed, and by providing mentoring and coaching as students undertake extended projects and develop presentations of finished work, both in and beyond the classroom.

The units in *Exploring Computer* Science contain individual lessons that taken together as a unit fit the construct for inquiry-based learning outlined in the following chart adapted from the “5 E Model”.

The Inquiry-Based Learning Cycle

(Adapted from the 5 E Model", R. Bybee)

Stage of Inquiry in an Inquiry-Based Science Program	Possible Student Behavior	Possible Teacher Strategy
Engage	Asks questions such as, Why did this happen? What do I already know about this? What can I find out about this? How can I solve this problem? Shows interest in the topic.	Creates interest. Generates curiosity. Raises questions and problems. Elicits responses that uncover student knowledge about the concept/topic.
Explore	Thinks creatively within the limits of the activity. Tests predictions and hypotheses. Forms new predictions and hypotheses. Tries alternatives to solve a problem and discusses them with others. Records observations and ideas. Suspends judgment. Tests ideas.	Encourages students to work together without direct instruction from the teacher. Observes and listens to students as they interact. Asks probing questions to redirect students' investigations when necessary. Provides time for students to puzzle through problems. Acts as a consultant for students.
Explain	Explains their thinking, ideas and possible solutions or answers to other students. Listens critically to other students' explanations. Questions other students' explanations. Listens to and tries to comprehend explanations offered by the teacher. Refers to previous activities. Uses recorded data in explanations.	Encourages students to explain concepts and definitions in their own words. Asks for justification (evidence) and clarification from students. Formally provides definitions, explanations, and new vocabulary. Uses students' previous experiences as the basis for explaining concepts.
Elaborate	Applies scientific concepts, labels, definitions, explanations, and skills in new, but similar situations. Uses previous information to ask questions, propose solutions, make decisions, design experiments. Draws reasonable conclusions from evidence. Records observations and explanations	Expects students to use vocabulary, definitions, and explanations provided previously in new context. Encourages students to apply the concepts and skills in new situations. Reminds students of alternative explanations. Refers students to alternative explanations.
	Checks for understanding among peers. Answers open-ended questions by using observations, evidence, and previously	Refers students to existing data and evidence and asks, What do you know? Why do you think...? Observes

Stage of Inquiry in an Inquiry-Based Science Program	Possible Student Behavior	Possible Teacher Strategy
Evaluate	accepted explanations. Demonstrates an understanding or knowledge of the concept or skill. Evaluates his or her own progress and knowledge. Asks related questions that would encourage future investigations.	students as they apply new concepts and skills. Assesses students' knowledge and/or skills. Looks for evidence that students have changed their thinking. Allows students to assess their learning and group process skills. Asks open-ended questions such as, Why do you think...? What evidence do you have? What do you know about the problem? How would you answer the question?

Concrete Instructional Strategies

There are several concrete instructional strategies that are included in each unit to implement this culturally relevant, student-centered, and inquiry-based vision.

- Each unit begins with a description of the topic, an explanation of the importance of this topic, possible social applications of this topic, and objectives/standards for the unit.
- Whenever possible, units begin with kinesthetic activity to get students involved in the unit topic. Students are more engaged when they go beyond seatwork to gain familiarity with the scope of a topic. Acting out computing concepts is one way to have students actively engaged in the curriculum.
- Whenever possible, units present the final unit project at the beginning of the unit so students understand what type of project they will engage in at the end of the unit. Daily assignments help scaffold their knowledge towards gaining the knowledge needed to complete a particular project. The final project represents a culmination of their new knowledge and provides an opportunity to expand their understandings to a particular socially-relevant problem.
- Computing terms and definitions are explicit and part of the instruction. The curriculum avoids unnecessary jargon which might distract from learning of the critical content. Students have opportunities to use writing to reinforce the literacy component behind these computing terms and definitions.
- Foundational computing topics are connected to the 'pop-technology' students have likely encountered: cellular phones, iPods, MySpace / Facebook, blogs, Internet browsing, etc.
- Whenever possible, real-world problems are presented in the context of socially-relevant issues impacting urban communities (housing, safety, poverty, health care, access to equal rights, educational opportunities, improving social services, translation services, transportation, etc.)

- Students have opportunities to work on problems that they help define and can individualize—i.e. selecting their own content for Web sites; creating original, not pre-scripted, problem-solving strategies, etc.
- Activities are designed to encourage students to work in a variety of collaborative settings: peer-programming, group research projects, etc. which encourage conversations around computing topics.
- Students will experience a variety of ways to communicate their answers—academic writing, writing a letter to a friend or companion, using presentation software, developing graphics or animation, listing algorithms, drawing illustrations, oral presentations, etc.
- Units incorporate examples of careers in computing as they arise in the curriculum. Students will be given hypothetical opportunities to act as a professional to take on the behavior and skills to solve a given problem.
- Although using technology is a core component of this curriculum, using computers is not necessarily embedded in the curriculum on a daily basis.

All of these strategies contribute to developing the problem-solving skills and algorithmic thinking processes that are emphasized throughout the course.

It is important to note that each unit focuses on different instructional strategies; this is purposeful. In some cases, it is because the particular subject matter lends itself more successfully to a particular set of strategies, but this was also done to highlight the wide variety of possible strategies that can be used effectively in teaching this course. We encourage teachers to experiment by trying strategies that work well for them in a variety of different places in the curriculum. Journal responses and blog entries can be used by students to communicate about their work in any of the units. Peer reviews, gallery walks, jigsawing, role plays and collaborative groups of varying sizes can be used for activities throughout the course. There are many other possibilities to consider.

Assessment

With the exception of the final projects, there are no specific assessments listed in the lesson plans. There are also very few specific “homework” assignments. Differences in grading policies, types of assessments required, and student schedules make it difficult to gauge the best combination of assessment tools to use in a particular environment. Teachers are encouraged to determine which class activities might lend themselves to some research outside of class and which might make useful assessments. Additional assessment instruments can be developed by individual teachers or teacher teams. All forms of assessment should meet the criteria outlined in the Nine Principles of Learning.

Overview of the Instructional Materials

The pages that follow contain the core of the materials teachers will need in order to plan and deliver *Exploring Computer Science.* The materials begin with the unifying themes and practices that are woven throughout the course followed by a Scope and Sequence chart that details the various topics included in the course, along with the unit in the course where each is introduced and reinforced. Teachers should continue to refer back to previous units where appropriate. For example, Unit 3 builds on many of the Unit 1 concepts by taking students from discussing and viewing websites to actually using and developing them. The approximate time allotment noted in the chart includes all activities from introduction through application.

Following the Scope and Sequence is an overview of each unit that includes the unit description and overall objectives of the unit. There is also a table that indicates the topics for each instructional day of the course.

Finally, are the daily lesson plans with detailed student activities and teaching strategies for each day. Each lesson has been built on a 55 minute class period. In schools where class periods are shorter or longer (or on varying block schedules) adjustments will need to be made; such adjustments may include combining lessons (for longer class periods) or assigning parts of the lesson for homework (for shorter class periods).

An attempt was made to provide enough detail to the teaching strategies sections to give teachers clear guidance as to the activities involved and the types of questions that might need to be asked to prompt discussion. At the same time, an effort was made not to be prescriptive.

Each unit includes supplementary materials, a final project, and a sample rubric for the final project.

Unifying Themes and Practices

The individual lessons in this course were developed to reinforce the unifying themes and support the use of the computational practices that we expect students to employ.

The three themes are

- The creative nature of computing
- Technology as a tool for solving problems
- The relevance of computer science and its impact on society

There are many technological tools that enable people to explore concepts and create exciting and personally relevant artifacts that impact society. In this course, programming is used as one of the tools, but not the only tool. Students are asked to be creative in designing and implementing solutions as they translate ideas into tangible forms. As students actively create, they will also discuss the broader implications of computing technologies.

Throughout the course students will gain experience in employing the following computational practices.

- Analyze the effects of developments in computing.
- Design and implement creative solutions and artifacts.
- Apply abstractions and models.
- Analyze their computational work and the work of others.
- Connect computation with other disciplines.
- Communicate thought processes and results.
- Work effectively in teams.

As students design and implement solutions using abstractions and models, they will analyze the processes they and their peers use to arrive at solutions, study the effects of their creations and learn how computing concepts connect explicitly and implicitly to other disciplines. Students will learn about the collaborative nature of computer science by working in teams and communicate the results of their work in writing and orally supported by graphs, visualizations and computational analysis.

Scope and Sequence

Topic	Focus	HCI	PS	WEB	PR	DA	ROB
1. Computers and the internet (~2 weeks)	1. Hardware components	I		R	R	A	A
	2. Software components	I		R	R	A	A
	3. Interaction of components	I		R	R	A	A
	4. Selection of appropriate components	I					
	5. Search engine fundamentals	I		R			
	6. Collaborative tools	I		R			
	7. Evaluating websites	I		R			
	8. Security on the Internet	I		R			
2. Models of intelligent behavior (~2 weeks)	1. What is intelligence?	I					
	2. Computers vs. humans	I	R	R	R	R	R
3. Algorithms and abstraction (~6 weeks)	1.Understanding the problem		I	R	R	A	A
	2. Exploring problems: problem-solving heuristics and strategies		I	R	R	A	A
	3. Design creation and representation		I	R	R	A	A
	4. Problem data		I	R	R	A	A
	5. Solution accuracy		I	R	R	A	A
	6. Design re-evaluation and refinement		I	R	R	A	A
	7. Decompose the complex		I	R	R	A	A
	8. Communicate results		I	R	R	A	A
	9. Algorithm efficiency		I		R	R	R
	10. Computationally intensive problems		I			R	R
	11. Unsolvable problem for a computer		I			R	R
	12. Computationally hard problems.		I			R	R
4. Connections between mathematics and computer science (~2 weeks)	1. Logic		I		R	A	A
	2. Binary number system		I				
	3. Basic Sets		I		R	A	A
	4. Concepts of functions		I		R	A	A
	5. De Morgan's laws		I		R	A	A
	6. Graphs		I		R	A	A
5. Creating computational artifacts (Web pages, programs, and robots) (~14 weeks)	1. Break a problem statement into specific requirements		I	R	R	R,A	R,A
	2. Design a solution to a problem		I	R	R	R,A	R,A
	3. Choose appropriate tools and techniques		I	R	R	R,A	R,A
	4. Code a solution from a design			I	R	R,A	R,A

Legend: I—Introduce, R—Reinforce, A—Apply

	5. Test a solution to identify errors		I	R	R	A	A
	6. Refine solution		I	R	R	A	A
	7. Documentation and justification		I	R	R	A	A
6. Data and information (~7 weeks)	1. Representation and storage	I	R		R	A	
	2. Methods for collection and generation	I	R			A	
	3. Patterns, trends, and discoveries	I	R			A	
	4. Evaluation		I			R,A	
	5. Computational models	I	R			A	
	6. Rapid testing		I			R,A	
7.Societal impacts of computing (weave throughout)	1. Fostering innovation						
	2. Legal and ethical concerns						
	3. Privacy and cyber security						
	4. Exploitation of information						
	5. Intellectual property						
	6. Limits on information access						
	7. Cultural influence						
	8. Equity, access, and power						
	9. Social and economic values						

Legend: I—Introduce, R—Reinforce, A—Apply

Overview Chart

Human Computer Interaction Unit Overview	
Instructional Day	**Topic**
1-2	Explore the concepts of *computer* and *computing*.
3-4	"Demystify" and learn the function of the parts of a personal computer. Learn the terminology of hardware components necessary for the purchase of a home computer.
5-7	Explore the world wide web and search engines. Experiment with a variety of search techniques, internet resources, and Web 2.0, applications. Evaluate websites.
8-9	Examine the implications of data on society and how computers are used for communications.
10	Tell a story with data.
11-14	Explore how computers are used as a tool for visualizing data, modeling and design, and art in the context of culturally situated design tools.
15-16	Introduce the concept of a computer program as a set of instructions.
17-19	Explore the idea of intelligence—especially as it relates to computers. Explore what it means for a machine to "learn". Discuss whether computers are intelligent or whether they only behave intelligently.
Problem Solving Unit Overview	
Instructional Day	**Topic**
1-2	Introduce data collection and problem solving.
3	Introduce the four steps of the problem solving process.
4-6	Apply the problem solving process. Use different strategies to plan and carry out the plan to solve several problems.
7-9	Reinforce the four steps of the problems solving process.
10-12	Count in the binary number system. Convert between binary and decimal numbers in the context of topics that are important to computer science.
13-14	Introduce the linear and binary search algorithms.
15-16	Explore sorted and unsorted lists and various sorting algorithms.
17	Introduce minimal spanning trees and how graphs can be used to help solve problems.
18-21	Final projects and presentations

Web Design Unit Overview	
Instructional Day	**Topic**
1-2	Explore issues of social responsibility in web use as well as the relative merits of the influence of the web on society, personal lives, and education.
3-4	Introduce the use of basic html.
5	Introduce basic formatting in html.
6-7	Explore image editing for the web using Photoshop or an image editor of choice.
8-10	Introduce basic css.
11-13	Explore the concept of separating style from structure by keeping separate html and css files.
14	Add hyperlinks to other websites.
15-16	Introduce a variety of page layout styles.
17-19	Practice the use of various design elements.
20-21	Introduce several different enhancements for website design, including web user interface elements combining Javascript, html, css, and Photoshop, accordion menus, lightbox and sliding images.
22-25	Final projects and gallery walk
Introduction to Programming Unit Overview	
Instructional Day	**Topic**
1	Introduce the Scratch programming language, including the basic terms utilized in the language.
2-3	Practice using the basic features of Scratch in the context of creating a simple program.
4	Create a dialogue between two sprites.
5-6	Introduce the methods of moving sprites in Scratch.
7-8	Practice the concept of event driven programming through the creation of an alphabet game.
9	Introduce the concept of broadcasting via role play.
10-13	Write Scratch stories and present them to the class. Conduct peer reviews.

14	Introduce the concept of variable.
15	Introduce the concept of conditionals.
16-17	Introduce And, Or and randomness.
18	Apply knowledge of conditionals to develop a Rock Paper Scissors program in Scratch.
19	Build on previous programming concepts to create a timer.
20-23	Create a timing game in Scratch and present it to the class. Peer reviews are conducted.
24	Investigate two types of games that may provide ideas for the final project.
25	Explain final project and the rubric for the final project.
26-28	Write Scratch programs for either My Community or Game project. Conduct peer reviews.
29	Complete final projects.
30	Presentations of final projects
Computing and Data Analysis Unit Overview	
Instructional Day	**Topic**
1	Review how data can be used for making a case/discovery and provide an overview of the final project.
2	Discuss photo ethics and student safety related to android phone use.
3-5	Distribute phones. Create groups. Discuss group roles and responsibilities. Navigate the android application. Navigate the online system.
6	Data check-in—Discuss issues that arise (aggregating data, etc.).
7-10	Introduce R/Deducer. Create maps using the latitude and longitude of a location and then create maps from a file of data.
11	Create maps with student data and related data set.
12-14	Discuss bar plots, categorical and continuous data, and mosaic plots as a vehicle for comparing categorical data, and looking at trends in data.
15	Create bar plots and mosaic plots with student data and related data set.
16-18	Review mean, median, minimum, maximum. Discuss various ways to subset data. Represent data with box plots and histograms.

19	Identify mean, median, minimum, maximum, create subsets, and create box plots and histograms with student data and related data set.
20-22	Use a variety of filters and queries to create subsets of text data. Create bar plots to graphically display the information.
23	Analyze text in student data and related data set.
24-26	Finalize data analysis for final project.
27-29	Develop website or Scratch program to present data analysis campaign.
30	Final project presentations

Robotics Unit Overview

Instructional Day	Topic
1	What is a robot? Identify the criteria that make an item a robot.
2-3	Evaluate robot body designs and create algorithms to control robot behavior.
4	Set up LEGO® Mindstorms® NXT® kit.
5	Build robot base.
6-7	Introduce the features of NXT Brick—the "brain" of the robot.
8-9	Introduce the features of the Mindstorms NXT software.
10-13	Program the robot using the Mindstorm Robot Educator Software tutorials.
14	Introduce RoboCup real life robotic competition and write instructions for tic-tac-toe.
15	RoboTic-Tac-Toe Tournament and introduction to RoboCupJunior Dance Challenge.
16-18	Build, program, and present a dancing robot.
19-23	Build program and present a rescue robot.
24-33	Final projects and presentations

Topic Descriptions and Objectives

Unit 1: Human Computer Interaction (4 weeks)

Topics to be addressed:

- Computers and the internet
- Models of Intelligent Behavior
- Societal impacts of computing

Topic Description:

In this unit students are introduced to the concepts of computer and computing while investigating the major components of computers and the suitability of these components for particular applications. Students will experiment with internet search techniques, explore a variety of websites and web applications and discuss issues of privacy and security. Fundamental notions of Human Computer Interaction (HCI) and ergonomics are introduced. Students will learn that "intelligent" machine behavior is not "magic" but is based on algorithms applied to useful representations of information, including large data sets. Students will learn the characteristics that make certain tasks easy or difficult for computers, and how these differ from those that humans characteristically find easy or difficult. Students will gain an appreciation for the many ways in which computing-enabled innovation have had an impact on society, as well as for the many different fields in which they are used. Connections among social, economical and cultural contexts will be discussed.

Objectives:

The student will be able to:

- Analyze the characteristics of hardware components to determine the applications for which they can be used.
- Use appropriate tools and methods to execute Internet searches which yield requested data.
- Evaluate the results of web searches and the reliability of information found on the Internet.
- Explain the differences between tasks that can and cannot be accomplished with a computer.
- Analyze the effects of computing on society within economic, social, and cultural contexts.
- Communicate legal and ethical concerns raised by computing innovation.
- Explain the implications of communication as data exchange.

Unit 2: Problem Solving (4 weeks)

Topics to be addressed:

- Algorithms and abstraction
- Connections between Mathematics and Computer Science
- Societal impacts of computing

Topic Description:

This unit provides students with opportunities to become "computational thinkers" by applying a variety of problem-solving techniques as they create solutions to problems that are situated in a variety of contexts. The range of contexts motivates the need for students to think abstractly and apply known algorithms where appropriate, but also create new algorithms. Analysis of various solutions and algorithms will highlight problems that are not easily solved by computer and for which there are no known solutions. This unit also focuses on the connections between mathematics and computer science. Students will be introduced to selected topics in discrete mathematics including Boolean logic, functions, graphs and the binary number system. Students are also introduced to searching and sorting algorithms and graphs.

Objectives:

The student will be able to:

- Name and explain the steps they use in solving a problem.
- Solve a problem by applying appropriate problem-solving techniques.
- Express a solution using standard design tools.
- Determine if a given algorithm successfully solves a stated problem.
- Create algorithms that meet specified objectives.
- Explain the connections between binary numbers and computers.
- Summarize the behavior of an algorithm.
- Compare the tradeoffs between different algorithms for solving the same problem.
- Explain the characteristics of problems that cannot be solved by an algorithm.

Unit 3: Web Design (5 weeks)

Topics to be addressed:

- Web page design and development
- Computers and the internet
- Algorithms and abstraction
- Societal impacts of computing

Topic Description:

This section prepares students to take the role of a developer by expanding their knowledge of algorithms, abstraction, and web page design and applying it to the creation of web pages and documentation for users and equipment. Students will explore issues of social responsibility in web use. They will learn to plan and code their web pages using a variety of techniques and check their sites for usability. Students learn to create user-friendly websites. Students will apply fundamental notions of Human Computer Interaction (HCI) and ergonomics.

Objectives:

The student will be able to:

- Create web pages to address specified objectives.
- Create web pages with a practical, personal, and/or societal purpose.
- Select appropriate techniques when creating web pages.
- Use abstraction to separate style from content in web page design and development.
- Describe the use of a website with appropriate documentation.

.

Unit 4: Introduction to Programming (6 weeks)

Topics to be addressed:

- Programming
- Algorithms and abstractions
- Connections between mathematics and computer science
- Societal impacts of computing

Topic Description:

Students are introduced to some basic issues associated with program design and development.
Students design algorithms and create programming solutions to a variety of computational problems using an iterative development process in Scratch. Programming problems include mathematical and logical concepts and a variety of programming constructs.

Objectives:

The student will be able to:

- Use appropriate algorithms to solve a problem.
- Design, code, test, and execute a program that corresponds to a set of specifications.
- Select appropriate programming structures.
- Locate and correct errors in a program.
- Explain how a particular program functions.
- Justify the correctness of a program.
- Create programs with practical, personal, and/or societal intent.

Unit 5: Computing and Data Analysis (6 weeks)

Topics to be addressed:

- Data and information
- Algorithms and abstraction
- Connections between mathematics and computer science
- Programming
- Societal impacts of computing

Topic Description:

In this unit students explore how computing has facilitated new methods of managing and interpreting data. Students will use computers to translate, process and visualize data in order to find patterns and test hypotheses. Students will work with a variety of large data sets that illustrate how widespread access to data and information facilitates identification of problems. Students will collect and generate their own data related to local community issues and discuss appropriate methods for data collection and aggregation of data necessary to support making a case or facilitating a discovery.

Objectives:

The student will be able to:

- Describe the features of appropriate data sets for specific problems.
- Apply a variety of analysis techniques to large data sets.
- Use computers to find patterns in data and test hypotheses about data.
- Compare different analysis techniques and discuss the tradeoffs among them.
- Justify conclusions drawn from data analysis.

.

Unit 6: Robotics (7 weeks)

Topics to be addressed:

- Robotics
- Algorithms and abstraction
- Connections between mathematics and computer science
- Programming
- Societal impacts of computing

Topic Description:

This unit introduces robotics as an advanced application of computer science that can be used to solve problems in a variety of settings from business to healthcare and how robotics enables innovation by automating processes that may be dangerous or otherwise problematic for humans. Students explore how to integrate hardware and software in order to solve problems. Students will see the effect of software and hardware design on the resulting product. Students will apply previously learned topics to the study of robotics.

Objectives:

The student will be able to:

- Identify the criteria that describe a robot and determine if something is a robot.
- Match the actions of the robot to the corresponding parts of the program.
- Build, code, and test a robot that solves a stated problem.
- Explain ways in which different hardware designs affect the function of a machine.
- Describe the tradeoffs among multiple ways to program a robot to achieve a goal.

The societal impacts of computing should be woven throughout the course.

Topic Description:

Throughout the course, is placed on how computing enables innovation in a variety of fields and the impacts that those innovations have on society. Computing is situated within economic, social and cultural contexts and, therefore, influences and is influenced by each of these. The proliferation of computers and networks raises a number of ethical issues. Technology has had both positive and negative impacts on human culture. Students will be able to identify ethical behavior and articulate both sides of ethical topics. Students study the responsibilities of software users and software developers with respect to intellectual property rights, software failures, and the piracy of software and other digital media. They are introduced to the concept of open-source software development and explore its implications. Students identify and describe careers in computing and careers that employ computing.

Objectives:

The student will be able to:

- Describe ways in which computing enables innovation.
- Discuss the ways in which innovations enabled by computing affect communication and problem solving.
- Analyze how computing influences and is influenced by the cultures for which they are designed and the cultures in which they are used.
- Analyze how social and economic values influence the design and development of computing innovations.
- Discuss issues of equity, access, and power in the context of computing resources.
- Communicate the legal an ethical concerns raised by computational innovations.
- Discuss privacy and security concerns related to computational innovations.
- Explain positive and negative effects of technological innovations on human culture.