Computer Science and Software Engineering Course Outline

Open doors in any career with computer science!

Students create apps for mobile devices, automate tasks in a variety of languages, find patterns in data, and interpret simulations. Students collaborate to create and present solutions that can improve people's lives.

How will computing and connectivity transform your world?

Computer Science and Software Engineering (CSE) is a new PLTW course being offered for the 2014-2015 school year. Students work in teams to develop computational thinking and solve problems. The course covers the College Board's new CS Principles framework. The course does not aim to teach mastery of a single programming language but aims instead to develop computational thinking, to generate excitement about the field of computing, and to introduce computational tools that foster creativity. The course also aims to build students' awareness of the tremendous demand for computer specialists and for professionals in all fields who have computational skills. Each unit focuses on one or more computationally intensive career paths. The course also aims to engage students to consider issues raised by the present and future societal impact of computing.

Students practice problem solving with structured activities and progress to open-ended projects and problems that require them to develop planning, documentation, communication, and other professional skills. Problems aim for ground-level entry with no ceiling so that all students can successfully engage the problems. Students with greater motivation, ability, or background knowledge will be challenged to work further.

The following is a summary of the units of study that are included in the course for the 2014-2015 academic year. The course is designed to cover all learning objectives in the College Board's 2013 draft CS Principles framework. In specific CSE projects and problems, students create artifacts and associated writing for CS Principles performance assessment tasks. Alignment with CS Principles Learning Objectives and with CSTA Level 3B Objectives is indicated in the PLTW CSE Curriculum Framework at the activity level. Alignment with NGSS, Common Core, and other standards will be available through the PLTW Alignment web-based tool. Activities, projects, and problems will be provided to the teacher in the form of student-ready handouts, teacher notes, and supplementary materials including code, instructional videos, and online practice questions as appropriate.

The course is planned for a rigorous pace, and it is likely to contain more material than a skilled teacher new to the course will be able to complete in the first iteration. Building enthusiasm for rigorous computer science among students is a primary goal of the course. Teachers are encouraged to emphasize content that will be fresh and exciting to students, and the course is structured to facilitate local adaptation to a particular group of students' prior knowledge and experience.

Unit 1   Algorithms, Graphics, and Graphical User Interfaces (48%)

Unit 2   The Internet (18%)

Unit 3   Raining Reigning Data (17%)

Unit 4   Intelligent Behavior (17%)

Unit 1: Algorithms, Graphics, and Graphical User Interfaces

The goal of Unit 1 is to excite students about programming and to build their algorithmic thinking and ability to use abstraction. Student creativity is emphasized as they work with Scratch™, App Inventor, and *Python*® to tell graphical stories, publish games and Android™ applications, and explore various development environments and programming techniques. Students will not only create original code, but also read and modify code provided from other sources. An Agile software development process is emphasized and personal and professional skills including collaborative skills take center stage. Students debate policy questions about the ownership and control of digital data and examine the implications for creative industries and consumers. In this unit students begin their exploration of career paths tied to computing.

Lesson 1.1	Algorithms and Agile Development
Lesson 1.2	Mobile App Design
Lesson 1.3	Algorithms in *Python*
Lesson 1.4	Images and Object-Oriented Libraries
Lesson 1.5	GUIs in *Python*

Lesson 1.1 Algorithms and Agile Development

The goal of this lesson is to introduce students to programming at a level appropriate to novice programmers. With an introduction to pair programming and the Agile software development process, students create original programs in Scratch that incorporate audio and visual elements while tackling algorithmic problems. The lesson opens with an introduction to how computing is affecting our lives. Students explore tools for collaboration over the Internet and select from these tools in order to manage the projects that they create. The foundations for later algorithmic thinking are built by focusing on the most common roles that variables fulfill, with an introduction to the conventions of object-oriented programming.

Lesson 1.2 Mobile App Design

The goal of this lesson is for students to build their skills by analyzing existing code, particularly with an emphasis on the roles of variables. Students create an Android app of their own design. The lesson begins with an introduction to binary representations of numbers, letters, colors, images, etc. using a CS unplugged activity in which students create a physical representation of data storage. Students work with and make minor modifications to two App Inventor programs, building their ability to analyze a complex program and incorporate event handlers into programs in meaningful ways. Students conclude by designing and creating their own Android app using pair programming and practicing the Agile software design process.

Lesson 1.3 Algorithms in *Python*

The goal of this lesson is for students to understand all information as bits and to transfer their understanding of algorithms to a new language, *Python*, which is powerful enough to raise all the opportunities and issues targeted in the course. Students are introduced to functional, imperative, and declarative programming paradigms with *Python*, again learning to use variables in the most common roles. Before learning about variable types and the fundamental algorithmic structures in *Python*, students simulate program execution in a model assembly language. After building strength with basic *Python* algorithms, students create algorithms to compete in a

round-robin tournament of the Prisoner's Dilemma, using the collaborative programming platform GitHub in the process.

Lesson 1.4 Images and Object-Oriented Libraries

The goal of this lesson is for students to become independent learners of a programming language, able to refer to documentation to use object-oriented libraries commonly available. The lesson begins with an unplugged activity to teach object-oriented concepts. Students build additional strength with *Python* algorithms, manipulating image files by modifying pixel data as well as using code libraries to work at higher levels of abstraction. As part of that work, they learn to use a variety of documentation including application-programming interfaces (APIs). Students read, discuss, and debate intellectual property issues associated with digital data. In the culminating problem of the lesson, they collaborate to create an image processing function that highlights the power of automation.

Lesson 1.5 GUIs in *Python*

The goal of this lesson is for students to conceive of any class of objects as an abstraction. Students will create a graphical user interface (GUI) with considerations of audience and accessibility. The lesson begins with an unplugged activity that generalizes the user interface topic of this lesson to the field of human-computer interaction. Students practice using an application-programming interface (API) to learn methods to affect an object's state. Students work with two APIs: the Tkinter Canvas for drawing and animation, and then the Tkinter toolbox of GUI widgets. Students are provided code for a simple GUI that implements a model-view-controller (MVC) pattern. Students will modify the elements of that pattern to suit their own needs. The lesson concludes with a problem in which students create a model-view-controller GUI using Scratch or *Python*. Strategies for documentation are reinforced, and Agile development is emphasized in the concluding problem.

Unit 2: The Internet

The goal of Unit 2 is for students to have a more concrete understanding of the Internet as a set of computers exchanging bits and the implications of these exchanges. Students use PHP and SQL to structure and access a database hosted on a remote server, learn how HTML and CSS direct the client computer to render a page, and experiment with JavaScript™ to provide dynamic content. The focus of the unit is on the protocols that allow the Internet to function securely to deliver social media and eCommerce content. Students work briefly in each of several Web languages to understand how the languages work together to deliver this content. The history and workings of the Internet are explored, and issues of security, privacy, and democracy are considered. Practical cyber security hygiene is included. Career paths in cyber security, web development, and information technology are highlighted.

Lesson 2.1   The Internet and The Web

Lesson 2.2   Shopping and Social on the Web

Lesson 2.3   Security and Cryptography

Lesson 2.1 The Internet and the Web

In this lesson, the goal is to build student understanding of the Internet as a set of computers exchanging bits in the form of packets. Students will learn to identify the components of their digital footprint. To provide a hook, students compare the designs, strengths, and weaknesses of their favorite web pages. In this context, students use an unplugged activity to understand (in broad brushstrokes) the content and flow of data when browsing the Web. They compare results from different search engines and learn to refine their search techniques. They review how to assess the trustworthiness of web-based media and consider the data flow that permits targeted advertisements. Students employ appropriate tools to explore the hierarchical nature of DNS and IP. Students identify ways that a web developer's decisions affect the user and ways that the user's decisions impact society. The tree structure of web documents is introduced alongside HTML and CSS. Paired key encryption and authentication are introduced with an unplugged activity.

Lesson 2.2 Shopping and Social on the Web

The goal for this lesson is for students to understand the role of client-side code, server-side code, and databases in delivering interactive web content. The hook is a problem in which CS students collaborate with art students to publish content on the Web. Students are provided with JavaScript and PHP code and can access an SQL database from a secure shell command line as well as through PHP. Students compare languages encountered so far to generalize the concepts of sequencing instructions, selection of instructions by conditionals, iteration, and the common roles of variables. Students explore and compare career paths within computing.

Lesson 2.3 Security and Cryptography

The goal of this lesson is for students to personally invest in maintaining online security and to improve their personal cyber security hygiene. Students focus on cyber security from the perspectives of the user, the software developer, the business, the nation, and the citizen. In the team competition at the end of the lesson, students explore parallel strands in encryption and security. Encryption is used as a route to explore the efficiency of algorithms and how the time for an algorithm to execute can depend on its input.

Unit 3: Raining Reigning Data

The goal of Unit 3 is for students to see the availability of large-scale data collection and analysis in every area they can imagine. Students examine very large data sets tied to themselves as well as to areas of work and society. They learn a variety of data visualization techniques and work to recognize opportunities to apply algorithmic thinking and automation when considering questions that have answers embedded in data. The complexity of the data sets, visualizations, and analysis increases in the second lesson of the unit, challenging students to generalize concepts developed in the first lesson.

Lesson 3.1   Visualizing Data

Lesson 3.2   Discovering Knowledge in Data

Lesson 3.1 Visualizing Data

The goal of this lesson is for students to be able to create visualizations to analyze sets of large data and to meaningfully interpret the patterns they uncover. They draw conclusions about themselves from relevant data, including local weather, the economics of their community, and naming trends with their name. At the beginning of the lesson, students weigh societal concerns around the collection and persistence of Big Data. The students learn how to use *Python* to make useful graphic representations of data, developing from familiar visualizations to more modern visual analyses like scaled-dot or colorized scatter plots and scatter plot arrays of multidimensional data sets. Students are introduced to basic Excel® spreadsheet programming and cell manipulation. In two activities at the end of the lesson, students deepen their understanding of the meaning of knowledge gained by visualization. Students use *Python* to add best-fit lines to scatter plots when the data suggest a linear relationship. A Monte Carlo simulation is used to help students appreciate the meaning of evidence for association between two variables.

Lesson 3.2 Discovering Knowledge from Data

As in the previous lesson, the goal of this lesson is for students to be able to create a range of visualizations to analyze complex sets of large data and to meaningfully interpret the patterns they uncover. The hooks are again conclusions they can draw about themselves from relevant data, including various geographic perspectives on their life and facial recognition of their own features. The lesson uses Excel as well as *Python* to manipulate and visualize data. Students examine geographic and social data using heat maps and directed graphs. Students experiment with object recognition and face recognition. They are challenged to discover clustering and association patterns lurking in data sets distributed across student computers and school sites, such that data cleaning and warehousing are necessary. Finally, student teams choose a question and answer it using large data.

Unit 4: Intelligent Behavior

In Unit 4, the emergence of intelligent behavior is explored from three distinct approaches: from single autonomous machines, from human crowd sourcing of data, and from separate algorithmic agents working in parallel. The goal is to galvanize the connections among computing concepts and between computing and society. The first lesson explores the hardware layer of computing, working from electrical breadboards to robotic units. Once students have implemented some robotic behaviors using physical components with human input, they automate behaviors by programming in C++. In the concluding lesson, students identify problems and questions that can be addressed with computer simulation, incorporating agent-based modeling. Students are challenged to explore the assumptions and parameters built into several simulations and to attach meaning to the results. Having explored a few applications of intelligent behavior emerging from algorithmic components, students reflect on the current and future state of artificial intelligence.

Lesson 4.1	Intelligent Machines
Lesson 4.2	Interpreting Simulations

Lesson 4.1 Intelligent Machines

The goal of Lesson 4.1 is to elucidate the inner workings of computing machinery for students at a hardware level. Students apply programming techniques learned previously to control Arduino robotic equipment. Students manipulate electronic components to implement specific robot behavior patterns. Programming is then used to adapt the robots' behavior to make intelligent decisions about when to "disobey" its operator. Students conclude the lesson with a programming challenge to implement fully autonomous behavior.

Lesson 4.2 Predicting, Understanding, and Communicating with Simulation

The goal of this lesson is to excite students about the range of questions that can be addressed with simulation and to build students' skill in using simulation to produce meaningful explanations, illustrations, or predictions. Students explore many agent-based GUI simulations in NetLogo. Students interpret results, modify the agents' behaviors by changing parameter values, and interpret the change in results. Students use Monte Carlo simulation when needed. The limit and future of predictive power of simulations are explored. The course ends with a return to the opening note: How will computation change our world?