3/26/2015

El Monte Union High School District

Course Outline

High School DISTRICT

Title: Computer Science & Software Engineering (PLTW)	This course meets graduation requirements:	Department/Cluster Approval	Date
		________________	__________
Transitional*_______(Eng. Dept. Only)	( ) English		
	( ) Fine Arts	________________	__________
	( ) Foreign Language		
Sheltered (SDAIE)*____Bilingual*___	( ) Health & Safety	________________	__________
	( ) Math		
AP**________Honors**__________	( ) Physical Education	________________	__________
	( ) Science		
Department: Industrial Technology	( ) Social Science	________________	__________
	(X) Elective (Math)		
Grade Level (s): 9-12		________________	__________
Semester___________Year X		________________	__________
Year of State Framework Adoption_____			

*Instructional materials appropriate for English Language Learners are required.

**For AP/Honors course attach a page describing how this course is above and beyond a regular course. Also, explain why this course is the equivalent of a college level class.

1. Prerequisite(s):

Introduction to Engineering Design (IED)
Principles of Engineering (POE)
Concurrently enrolled in Integrated Math 2 or above

2. Short description of course which may also be used in the registration manual:

Students work in teams to develop computational thinking and problem solving skills. 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 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.

3. **Describe how this course integrates the schools ESLRs (Expected School-wide Learning Results):**

This course integrates the ESLRs by combining elements of hands on career technical education in various areas while using the skills of reading, writing and mathematics. Computer-based programming technology will be used. Each unit in the curriculum focuses on one or more computationally intensive career paths.

4. **Describe the additional efforts/teaching techniques/methodology to be used to meet the needs of English Language Learners:**

Additional efforts, techniques and methodologies used to meet the needs of ELL students include the pairing or partnering of ELL students with bilingual advanced students open to assisting struggling or challenged students. Additionally, when available, ELL students will be provided with learning materials in their native language. Furthermore, if available, instructional aides will be provided to translate and assist ELL students with coursework. Lastly, since the course has day to day objectives in place, SIOP model practices will be implemented, such as posting daily learning objectives in written form for the entire class to see.

5. **Describe the interdepartmental articulation process for this course:**

Skills developed in IED and POE will continue to be used in CSE. Students will develop computational thinking and problem skills that require them to develop planning, documentation, communication, and other professional skills. These target skills and involved assignments will be shared and discussed with other departments, including math and science courses, with the goal of increasing student performance in the involved departments.

6. **Describe how this course will integrate academic and vocational concepts, possibly through connecting activities. Describe how this course will address work-based learning/school to career concepts:**

The course 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 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.

7. **Materials of Instruction (Note: Materials of instruction for English Language Learners are required and should be listed below.)**

A. Textbook(s) and Core Reading(s):

As required by the PLTW materials guide (updated yearly).

B. Supplemental Materials and Resources:

As required by the PLTW materials guide (updated yearly). 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.

C. Tools, Equipment, Technology, Manipulatives, Audio-Visual:

As required by the PLTW materials guide (updated yearly).

8.

- **Objectives of Course**

Building enthusiasm for rigorous computer science among students is a primary goal of the course. The course aims to develop computational thinking, to generate excitement about the field of computing, and to introduce computational tools that foster creativity, build students' awareness of the tremendous demand for computer specialists and for professionals in all fields who have computational skills, and to engage students to consider issues raised by the present and future societal impact of computing.

- **Unit detail including projects and activities including duration of units (pacing plan)**

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® programming languages to tell graphical stories, publish games and Android™ applications, and explore various development environments and programming techniques. Students will create original code and read and modify code provided from other sources. An Agile software development process is emphasized and personal, professional, and 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.

- 1.1 Algorithms and Agile Development Lesson (10 days)
- 1.2 Mobile App Design Lesson (12 days)
- 1.3 Algorithms in Python Lesson (14 days)
- 1.4 Images and Object-Oriented Libraries Lesson (15 days)
- 1.5 GUIs in Python (14 days)

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 and 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 acquire methods that 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.

- 2.1 The Internet and the Web (10 days)
- 2.2 Shopping and Social on the Web (12 days)
- 2.3 Security and Cryptography (14 days)

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 be dependent 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.

- 3.1 Visualizing Data (20 days)
- 3.2 Discovering Knowledge from Data (14 days)

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 of multidimensional data sets. Students are introduced to basic Excel® spreadsheet programming and cell manipulation. 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. Students use statistics to deepen the meaning of knowledge gained by visualization. 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 multidimensional data sets using scatter plot arrays and view 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 linear correlation 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 two distinct approaches: 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 discrete components to integrated circuits. The exponential advancement of electronics, low on the ladder of abstraction, is connected to advancements at the highest levels on the ladder of abstraction, where artificial intelligence and simulation and modeling are impacting all fields. 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.

- 4.1 Moore's Law and Modeling (14 days)
- 4.2 Intelligent Agents (20 days)

Lesson 4.1 Moore's Law and Modeling

In this lesson, students construct an understanding of how the explosion of technology over the last two decades has impacted every realm of study and employment. Students begin by researching the impact of computer modeling and simulation which have been made possible by the rapid increase in computational power due to the continued applicability of Moore's Law. They then manipulate discrete electronic components to create logic gates and create comparable results using integrated circuits to get a feel for what it means to double the number of transistors that can fit in a given area. Students explore simulation in NetLogo directly by manipulating a model of predation and a model of the spread of viruses in humans. The lesson concludes with an examination of the code of ethics for simulationists and reflection on the necessity of adhering to such a code.

Lesson 4.2 Intelligent Agents

In this lesson, students experiment with materials designed to illuminate the rise of intelligent and complex behavior from simple rules and seemingly unintelligent agents. Students begin by studying a model of Langton's ant, a simple Turing machine with some surprising emergent behavior. The students manipulate models of neurons and neural networks. Students design and conduct their own experiments on a model of their own choosing using Monte Carlo methods. Students explore the generation and observation of fractals and study a diffusion limited aggregation model for producing fractal behavior. In the final project of the course, students choose a tool or tools that they have learned about in the course and apply their knowledge to create a novel product of their own design. They present their product to the class along with reflections about how it is tied to everything they've learned about computer science.

- **Indicate references to state framework(s)/standards (If state standard is not applicable**

then national standard should be used)

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.

- **Student performance standards**

The Common Core Standards, including the standards for Mathematical Practice, will be s implemented. The following is the grading scale used for overall performance:

A = 90%-100%
B = 80%-89%
C = 70%=79% 7
D = 60-69%
F = Below 60%

- **Evaluation/assessment/rubrics**

Through a balanced approach, assessment is an ongoing activity. Students demonstrate their knowledge throughout the course by completing activities, projects, and problems using a variety of assessment tools, such as performance rubrics and reflective questioning to deepen and expand their knowledge and skills.
PLTW's assessment experts apply industry best practices and methods to design, test, and implement End of Course (EoC) assessments. The EoC assessments will result in valid and reliable scores on overall student performance within the course. The EoC assessment gives students an objective evaluation of their achievement, and stakeholders obtain data to make informed decisions.

- **Include minimal attainment for student to pass course**

Overall grade percentage of 70% resulting from activities, projects, and problems using a variety of assessment tools, such as performance rubrics and reflective questioning.