Background

Advanced Placement™ Computer Science (AP CS) Principles™ is a proposed AP course that seeks to broaden participation in computing and CS. The CS Principles Framework includes computational thinking, practices and Big Ideas, Key Concepts, and Supporting Concepts. There have been 20 early adopters and there are four official pilot sites (2012–2013) and many more unofficial pilot sites led by innovative CS educators. The College Board™ is not in the business of publishing a recommended curriculum to meet course objectives. Educators interested in teaching the course have only the goals and objectives defined by the College Board and documents from previous pilots to work from.

Relevant and accessible contexts and technologies are essential to reaching students. The CS Principles course objectives provide a fertile foundation for exploring different technologies in a wide variety of situations. One of the hallmarks of the CS Principles exam (debuting in the 2016-2017 school year) is that it will include portfolio assessments and will be programming language agnostic; teachers will be able to use any programming language that supports the computing concepts and objectives of the course.

This curriculum offers multiple collaborative projects using current technologies such as Microsoft™ Office™ Web Apps and SkyDrive™ for productive collaboration, Microsoft Excel™ for data analysis and visualization, Kodu™ Game Lab for game development, and the approachable programming language, Small Basic™, a more traditional programming language that provides a transition to higher level languages like Java and C#.

CS Principles is a course that is meant to appeal to females and underrepresented minorities. The CS Principles: Computation in Action curriculum modules are ideal for the classroom and can be adapted to support a variety of gender and ethnic equity programs charged with increasing STEM and CS interest.

The CS Principles: Computation in Action curriculum supports the CSTA K–12 Computer Science Standards level 3 courses with advanced topics in exploring real-world problems by applying computational thinking to develop solutions. Projects and learning activities emphasize algorithmic problem solving, team collaboration, ethical computing, and modern collaborative tools and technologies.

The new Next Generation Science Standards (NGSS™), the next evolution in National Science Standards™, include computational thinking standards. The same computational thinking practices...
underlie the CS Principles: Computation in Action curriculum. The modules in CS Principles: Computation in Action are inherently cross-disciplinary and can also be adapted to specifically target the NGSS and teach computational thinking lessons in the science classroom or in cross-curricular projects.

CSTA recognizes and thanks Microsoft for generously supporting the development of this curriculum.

Introduction
CS Principles: Computation in Action engages students in socially-relevant, project-based learning activities designed to foster computational thinking within the Big Ideas and Concepts of the AP CS Principles Framework: Creativity, Abstraction, Data, Algorithms, Programming, Internet, and Impact. Students will put the principles of CS into action by creating interdisciplinary computational artifacts that combine CS with music, art, literature, and science (life, physical, and social). AP CS Principles promises exciting opportunities for CS educators and students for broadening participation with up-to-date technologies and relevant content. Invigorate your school’s CS program with the CS Principles: Computation in Action curriculum. This curriculum package includes lesson plans, activities, assessments, and video tutorials.

CS Principles: Computation in Action will engage students and encourage them to continue in the CS and IT course pathways. The content of CS Principles: Computation in Action is interesting, approachable, relatable, current, inherently interesting, and broadly appealing.

Unit 1: Algorithms
Algorithms are everywhere! They are responsible for music and movie recommendations on media player websites, product recommendations on commerce sites; they even recommend friends on social media sites. They govern financial services and control a significant percentage of all stock trading on Wall Street. We create algorithms (strategies) every time we play a card game or board game. In Unit 1, students will examine algorithms in everyday life, from maneuvers involved in dancing and sports to programs that control actions in video games. They will learn about types of algorithmic instructions, how to evaluate the quality of an algorithm, and how to express algorithms to solve computational problems. The unit culminates with a cross-curricular game-design project in which students will model algorithms in natural language, pseudo-code, and the Kodu Game Lab visual programming language.

Unit 2: Programming as an Expression of Creativity
In Unit 2 students will advance from examining and creating stand-alone algorithms to incorporating multiple algorithms into complete programs with user input and graphical output. Small Basic is used as the programming tool due to the approachability of the language and the simplicity of the editing environment. Students will learn how to work collaboratively and effectively in a programming environment by making use of pair programming and agile programming methodologies. The labs have multiple versions of varying degrees of difficulty, allowing teachers to scaffold and individualize learning.
to students’ needs. Near the beginning of the unit, students will be working with nearly complete programs in which most of the code is provided so that they can concentrate on particular concepts. They will eventually progress to writing the majority of the code themselves. At the end of this unit, students will collaboratively code a program or series of programs to solve a problem of interest to the students and reflect on the teamwork experience, development process, and the code itself. They will work in teams as designers, programmers, artists, and managers to design, develop, test, and refine their program.

**Unit 3: Data and Abstraction**

**Unit 3** begins with students learning how a computational system is composed of a hierarchy of abstractions that at its root, is controlled by the nature of digital information. Students will also create abstractions to represent the structure of data and the computational processes that act on data. They will use their algorithmic and programming skills to analyze data sets and abstract data. Students will learn how to analyze problems and create models and simulations to not only answer questions, but also propose additional investigations. The unit culminates with a collaborative project in which students select a large data set of interest, investigate a problem, justify their choice of computational tools, and create a digital artifact to report their results.

**Unit 4: The Internet and Impact**

In **Unit 4**, students will learn about the technology of the Internet and how computing enables impact on a global scale and innovation in a variety of fields. Students will explore a variety of case studies starting with the invention of the Internet, the numerous technological products that the Internet has enabled, and the significant societal changes that have resulted. To conclude Unit 4, students will imagine a world where technology solves the toughest problems. They will work collaboratively to connect their understanding of the Internet and the global impact of computing to research a specific contemporary problem that is associated with the United Nations Millennium Development Goals, propose a computational solution, and create a digital prototype of their solution.
Standards


Curriculum Structure

Document Organization: At the highest layer is a folder for each unit. In each Unit folder, you will find a series of lesson folders and a curriculum framework for that unit. For example, the Unit 1 Algorithms folder contains folders for lessons 1.1 to 1.6 and the Unit 1 Curriculum framework document. Each lesson folder contains a central lesson plan document which may last a few to several days. The lesson plan document will refer to worksheets, tests, quizzes, and projects that are located in the same folder. These documents will all start with the lesson number, the type of document it is, and then the title of the document. For example, “1.1 Lesson Plan – Introduction to Computational Thinking” Note: the numbering system described above is not representative of the folder structure in these preview documents but will be in the completed curriculum available in the fall of 2013.

The Lesson Plan document for each unit includes a detailed timeline and description of the labs and activities. Class period is defined as a 50-minute session.

Advanced Placement (AP)® is a registered trademark of the College Board and the National Merit Scholarship Corporation, which were not involved in the production of, and do not endorse, this product.
Lesson plan structure:

1. Unit number, lesson number, number of expected instructional days
2. Lesson overview
3. Objectives
4. Outline and time allocation
5. Resources
6. Lesson instructions

How to use the curriculum documents:

1. Open a unit folder.
2. Select a lesson folder.
3. Read the Lesson Plan document.
4. Prepare the associated resources.
5. Execute the lesson.

Content Coverage: The AP CS Principles Learning Objectives and Evidence Statements are grouped into 7 Big Ideas. While the interdependent relationship between the 7 Big Ideas is reflected in how their associated objectives are distributed throughout the curriculum, the objectives of each Big Idea are concentrated within one of the 4 units of this curriculum.

While CS Principles: Computation in Action curriculum is intended to be used in a year-long course, it was not written with the purpose of providing daily lesson plans for the entire year. Rather, teachers are provided with lesson materials for key topics in each unit, as well as a detailed culminating project plan. Where noted in the lesson plans, teachers are expected to create new material, use their own existing material, adapt lessons from other curriculum sources, or find additional resources to supplement the curriculum.

Technical notes

Software requirements:

1. Download and install all required software:
   a. Small Basic (smallbasic.com)
   b. Small Basic Extensions (http://extendsmallbasic.codeplex.com/)
      i. Unzip SmallBasicFun.zip, two files will be created.
      ii. Create a folder named 'lib' in C:\Program Files\Microsoft\SmallBasic
      iii. Copy SmallBasicFun.dll and SmallBasicFun.xml into the folder named 'lib'
   c. Kodu Game Lab (www.kodugamelab.com)


Hardware requirements:

Advanced Placement (AP)® is a registered trademark of the College Board and the National Merit Scholarship Corporation, which were not involved in the production of, and do not endorse, this product.
1. Suggested: PC running Windows Vista/Windows 7/Windows 8 with latest Service Packs with Internet access
2. Minimum: PC running Windows XP with latest Service Pack

**Resources and Web Links:**

Kodu Game Lab: kodugamelab.com

Microsoft Small Basic: smallbasic.com

Small Basic Extensions: extendsmallbasic.codeplex.com/


* This textbook is available at no cost from www.bitsbook.com/excerpts/ and is distributed under a Creative Commons License.

**Pedagogy**

**Prerequisite skills:** This high school course requires successful completion of Algebra.

**Sample Mastery Objectives/Results:**

**Unit 1: Algorithms**

- Given a natural language procedure, label examples of sequencing, selection, iteration, and recursion.
- Collaborate with others to create a complex algorithm composed of several simpler algorithms.
- Analyze the advantages and disadvantages of representing the same algorithm in a natural language, pseudo-code, a visual, and a textual programming language.
- Create a language to represent an algorithm in an area of knowledge such as a sport or hobby. Give examples of the algorithm and have other students use the language to represent other algorithms related to the sport.
- Identify common characteristics of problems that can’t be solved using an algorithm.
- Compare and rate/rank the efficiency, correctness, and clarity of algorithms designed to solve the same task.
- Explain how an algorithm works by writing inline comments in the code for an algorithm written in a visual or textual programming language.
Unit 2: Programming as an Expression of Creativity

- Describe the editing, compiling, and execution of code.
- Create functions and reuse them.
- Describe the process and advantages of using a library of code or API in creating more complex programs.
- Identify infringements on style conventions, syntax errors, and logic errors in code and correct them.
- Use code comments to describe variables and algorithms in programs, communicate with collaborators, understand another programmer’s code, identify misconceptions, and correct errors.
- Translate Boolean expressions into coded conditions for selection structures.
- Create a computational artifact that reflects the use of algorithms and programming to create a game that delivers a message.

Unit 3: Data and Abstraction

- Collect data and represent it in a form that can be computed.
- Convert an image into its numeric representation, transform the image, and output the resulting image.
- Abstract the concept and workings of a number system to represent a decimal number in an unknown base.
- Describe the abstractions that represent digital data and the data are stored physically.
- Explain the differences between analog and binary data and give examples of each.
- Process raw data and explain the benefits and negative consequences.
- Describe an algorithm used to build knowledge from a data set and the advantages of analyzing the data set computationally.

Unit 4: The Internet and Impact

- Describe how the Internet functions at multiple levels of detail by identifying the abstractions used at a higher level, the protocols and systems used to communicate information, and the programs that facilitate the communication.
- Create a Hypertext Markup Language (HTML) document with text and graphics and evaluate it for compliance with current web standards.
- Describe the evolving standards used on the Internet to represent addresses and names.
- Diagram a network topology to demonstrate how a message from one mobile device can be received by another.

Advanced Placement (AP)® is a registered trademark of the College Board and the National Merit Scholarship Corporation, which were not involved in the production of, and do not endorse, this product.
- Describe the software, hardware, and human components involved in implementing cyber security.
- Relate innovations in computing and the Internet to societal changes, both positive and negative.
- Identify a societal problem related to one of the United Nations Millennium Development goals, propose a computational solution, and create a digital prototype of a solution.