PLTW Computer Science
Computer Science Essentials | Course Outline

Collaborate to solve problems and create value for others through innovation and creativity. Through programming mobile apps, self-driving vehicles, and authentic day-to-day solutions, students learn computational thinking skills and put their designs into practice.
Whether these are your first steps in computer science, or a continuation of your journey, Computer Science Essentials will give you confidence to succeed today and beyond.

Computer Science Essentials (CSE) is designed to be a full-year (160-day) course implemented in the 9th grade. This course is an excellent entry point for new high school computer science (CS) learners. And students who have prior CS experiences will find ample opportunity to expand upon those experiences in this course. All students who take CSE will have many opportunities for creative expression and exploration in topics of personal interest, whether it be through app development or connecting computing with the physical world.

CSE is designed with strong connections to the Computer Science K12 Frameworks (CS K12), the Computer Science Teachers Association K-12 Computer Science (CSTA K-12 CS) Level 3A Standards, and the Standards for Technological and Engineering Literacy (STEL). Though CSE is not an instance of the AP Computer Science Principles course (CSP), it will boost student success for those who continue in the PLTW CSP course. These intentional connections to widely accepted standards will help students gain confidence and reinforce essential concepts and skills that build toward life-long success in the computer science pathways beyond just PLTW courses.

PLTW CSE introduces students to coding fundamentals through an approachable, block-based programming language where they will have early success in creating usable apps. As students sharpen their computational thinking skills, they will transition to programming environments that reinforce coding fundamentals by displaying block programming and text-based programming side-by-side. Finally, students will learn the power of text-based programming as they are introduced to the Python® programming language.

The course engages students in computational thinking practices and collaboration strategies, as well as industry-standard tools authentic to how computer science professionals work. Students will learn about professional opportunities in computer science and how computing can be an integral part of all careers today. The following is a list of the units of study in the course. (Note: Targeted percentages may shift.)

Unit 1	Creative Computing: Building with Blocks	(25%)
Unit 2	Computing and Society: Transitions to Text	(25%)
Unit 3	Solving with Syntax	(35%)
Unit 4	Computing with a Purpose	(15%)

Unit 1: Creative Computing: Building with Blocks (45 days)

Unit 1 welcomes new and returning students to the world of computer science and coding fundamentals. Students work with MIT App Inventor to create basic apps that rely on the concepts of event-driven programming, branching, iteration, variables, and abstraction—the building blocks of creating with code. Students are introduced to essential computational thinking practices, such as developing abstractions, collaborating around computing, and communicating as they create, test, and refine computational artifacts of Android™ apps.

Creative Computing: Building with Blocks Unit Summary

Lesson 1.1	Introduction to Computer Science Essentials	(19 days)
Lesson 1.2	Collaborating Around Computing	(17 days)
Lesson 1.3	Innovation and Problem Solving	(9 days)

Lesson 1.1 Introduction to Computer Science Essentials

Mobile computing has changed our world, and many of today's students have never known a life without apps. This lesson gives students the tools they need to create their own apps using MIT App Inventor. The goal of this lesson is to introduce students to coding fundamentals through block-based programming. Students will develop independent and collaborative strategies that will help them communicate around computing as they learn and reinforce the fundamental concepts of coding. With a powerful yet approachable tool, students will use their creativity to produce computational artifacts like those that are essential to all of us today.

Activity 1.1.1	Getting Started with Block-Based Programming: Digital Doodle	(3 days)
Activity 1.1.2	Algorithms and Coding Fundamentals: Happy Accelerometer	(3 days)
Activity 1.1.3	Conditionals and Event-Driven Programming: Happy Balance	(2 days)
Activity 1.1.4	Local and Global Variables: Guessing Game 2 Player	(3 days)
Activity 1.1.5	Iteration and Loops: Guessing Game 1 Player	(3 days)
Project 1.1.6	App Development: Creative Expression	(5 days)

Lesson 1.2 Collaborating Around Computing

This lesson focuses on collaborative strategies that coding professionals use when creating programs and applications, while it continues to introduce essential concepts in computer science and coding. The lesson also introduces the idea that computer science can be more than just innovation and creative expression; it can be powerful in trying to solve many problems in today's world. Students apply an Agile development process and task decomposition to solve a problem that meets the needs of others.

Activity 1.2.1	Problem Solving: Interview Database	(2 days)
Activity 1.2.2	Algorithms and APIs: Hack Attack	(3 days)
Activity 1.2.3	Procedural Abstraction: Price per Slice	(3 days)
Activity 1.2.4	Lists: Survey Says	(3 days)
Project 1.2.5	App Development: Problem Solving and Innovation	(6 days)

Lesson 1.3 Innovation and Problem Solving

The final lesson of this unit gives students the freedom to select the focus of their development in choosing the type of app they would like to collaborate to create. Student groups will apply development strategies and user-centered research to create an app that has value to others. Students will gain insight on the importance of creativity, persistence, and value of diverse perspectives in an iterative development process.

Problem 1.3.1	App Development: Creating Value for Others

Unit 2: Computing and Society: Transitions to Text (45 days)

Unit 2 continues to reinforce coding fundamentals as students are gradually introduced to text-based programming. In this unit, students will explore the impacts of computer science on our society and bring coding off the screen and into the physical world. Students will learn how images can be used to make decisions in programs and explore real-world applications and innovations that will shape our future.

Computing and Society: Transitions to Text

Lesson 2.1	Transitions to Text-based Coding (16 days)
Lesson 2.2	Computing and Careers in our Society (19 days)
Lesson 2.3	Computing in Our World (10 days)

Lesson 2.1 Transitions to Text Based Coding

Block-based programming is a great way to introduce coding fundamentals, but many students want to know, "What is happening inside those blocks?" Lesson 2.1 introduces students to the idea of a lower level of abstraction in a programming language. Students will develop in an environment that allows them to create in blocks, but see that same code in a text-based language.

Activity 2.1.1	Transitioning from Blocks to Text	(2 days)
Activity 2.1.2	Dead Reckoning	(3 days)
Activity 2.1.3	Coding Fundamentals: Lists	(3 days)
Activity 2.1.4	Coding Fundamentals: 2D Lists	(4 days)
Project 2.1.5	Map it, Drive it	(5 days)

Lesson 2.2 Computing and Careers in Our Society

Just as clicks of a button or "swipes" of a screen are used to trigger events in an app, today, images are becoming increasingly important as a way to make decisions in programming. In this lesson, students will explore image processing and other innovations that are changing our society. Students will also begin to investigate the wide range of careers in computer science and how computational thinking is an important part of the majority of professions today and in the future.

Activity 2.2.1	Careers, Innovation, and Ethics in Computer Science	(4 days)
Activity 2.2.2	Image Processing: Identification	(4 days)
Activity 2.2.3	Decisions from Images	(4 days)
Project 2.2.4	Image Processing: Navigation and Collision Avoidance	(7 days)

Lesson 2.3 Computing in Our World

Tomorrow's solutions involve all of us. In the final lesson, student groups will learn how to take collaborations to scale to achieve a common goal.

Project 2.3.1	Create Performance Task: Cooperative Driving	(10 days)

Unit 3 Solving with Syntax (55 days)

The goal of Unit 3 is for students to begin to understand and use the flexibility and power of programming in a text-based environment. In this unit, students will continue to build on coding fundamentals as they apply the same coding concepts, computational thinking practices, and development processes introduced in units 1 and 2.

Solving with Syntax Unit Summary

Lesson 3.1	Text-based Coding	(9 days)
Lesson 3.2	Text-based Solutions	(30 days)
Lesson 3.3	The Power of Text-based Programming	(16 days)

Lesson 3.1 Text-Based Coding

In this lesson, students will reinforce previously learned concepts as they are introduced to the power of programming in a text-based language. The goal of this lesson is for students to become comfortable implementing algorithms using conditionals and loops in Python.

Activity 3.1.1	Python Programing	(1 day)
Activity 3.1.2	Variables and Conditionals	(3 days)
Project 3.1.3	Combo Menu	(5 days)

Lesson 3.2 Text-based Solutions

In this lesson, students will continue to explore the use of text-based programming. The lesson ends with students creating a game simulation that allows them to make generalizations and develop functions that attempt to detect and react to another team's strategy.

Activity 3.2.1	Lists and Elements	(4 days)
Activity 3.2.2	Social Media: Posting Objects	(6 days)
Activity 3.2.3	Iteration and Counts	(5 days)
Activity 3.2.4	Course Registration: Functions	(7 days)
Project 3.2.5	Artificial Intelligence: Rock, Paper, Scissors Simulation	(8 days)

Lesson 3.3 The Power of Text-based Programming

In the final unit of this lesson, students will work with a team to create a program that automates the solution of a problem from one of their other classes.

Problem 3.3.1	From Paper to Programming	(16 days)

Unit 4 Computing with a Purpose (35 days)

The final unit in CSE allows students to apply all that they have learned in a student-defined, student-driven development. Whether creating an app, a website, or a physical computing device, students will apply computational thinking practices and a strategic development process to create computational artifacts that solve problems and create value for others. Students will collaborate the way computing professionals do as they pursue solutions to authentic needs. For those students continuing to PLTW CSP, this unit provides an excellent model of how to participate in, document, and create a performance task for AP CSP.

Computing with a Purpose Unit Summary

Lesson 4.1	Innovation of Computational Problem Solving	(35 days)

Lesson 4.1 Innovation of Computational Problem Solving

The goal of this lesson is to allow students the opportunity to apply the collaboration, technical, and communication skills that they have developed to solve an authentic problem that is relevant to them.

Problem 4.1.1	Create Performance Task: Your Development	
Part A	Find an Idea to Pursue	(4 days)
Part B	Set Your Development Milestones	(1 day)
Part C	Prepare, Investigate, and Plan	(3 days)
Part D	Design, Create, and Test	(15 days)
Part E	Evaluate and Reflect	(7 days)
Part F	Present	(5 days)