Computer Science Essentials

PLTW Framework

Course Level

PLTW Framework - Overview

PLTW Frameworks are representations of the knowledge, skills, and understandings that empower students to thrive in an evolving world. The PLTW Frameworks define the scope of learning and instruction within the PLTW curricula. The framework structure is organized by four levels of understanding that build upon each other: Knowledge and Skills, Objectives, Domains, and Competencies.

The most fundamental level of learning is defined by course Knowledge and Skills statements. Each Knowledge and Skills statement reflects specifically what students will know and be able to do after they've had the opportunity to learn the course content. Students apply Knowledge and Skills to achieve learning Objectives, which are skills that directly relate to the workplace or applied academic settings. Objectives are organized by higher-level Domains.

Domains are areas of in-demand expertise that an employer in a specific field may seek; they are key understandings and long-term takeaways that go beyond factual knowledge into broader, conceptual comprehension.

At the highest level, Competencies are general characterizations of the transportable skills that benefit students in various professional and academic pursuits. As a whole, the PLTW Frameworks illustrate the deep and relevant learning opportunities students experience from PLTW courses and demonstrate how the courses prepare students for life, not just the next grade level.

To thrive in an evolving world, students need skills that will benefit them regardless of the career path they choose. PLTW Frameworks are organized to showcase alignment to in-demand, transportable skills. This alignment ensures that students learn skills that are increasingly important in the rapidly advancing, innovative workplace.

Competencies (C), Domains (D), Objectives (O), Knowledge and Skills (KS)

C1 Creativity and Problem Solving

Computing enables people to use creative development processes to create computational artifacts for creative expression or to solve a problem.

D1 Creativity

Computing is a creative activity. Creativity and computing are prominent forces in innovation; the innovations enabled by computing have had and will continue to have far-reaching impact.

O1.1 Apply a creative development process when creating computational artifacts. LO 1.1.1 [P2]

KS1.1.1 Translate ideas into tangible form by creating computational artifacts and employing an iterative and exploratory process. EK 1.1.1B

O1.2 (1V) Create a computational artifact for creative expression. LO 1.2.1 [P2]

KS1.2.1 Identify a computational artifact as something created by a human using a computer and differentiate between a program, an image, an audio, a video, a presentation, or a web page file. EK 1.2.1A

KS1.2.2 Discuss how creativity, collaboration, and curiosity can lead to innovation.

D2 Problem-Solving Mindset

There are professional characteristics and habits of action that help people create value for society through innovation and problem solving.

O2.1 Describe moments within a process where curiosity, persistence, and the positive aspect of failure played an important role in gaining understanding about a problem or unexpected observation.

KS2.1.1 (2b IWR) Describe difficulties and/or opportunities you encountered and how they were resolved or incorporated.

O2.2 Engage stakeholders in a problem and use their perspectives to shape the course of your development.

KS2.2.1 Identifying programmer and user concerns that affect the solution to problems. EK 5.1.2G

KS2.2.2 Consult and communicate with program users in program development to solve problems. EK 5.1.2H

D3 Problem-Solving Process

A computational problem solving process is an iterative, systematic approach by which a team generates and validates a proposed solution.

O3.1 Apply and describe an iterative process based on user-centered research to solve a problem.

KS3.1.1 Apply and describe an iterative process used during the development of a solution.

KS3.1.2 Use user-centered research and design techniques to create software solutions. 3A-A-5-5

O3.2 Identify and apply decomposition as a critical step in problem solving.

KS3.2.1 Deconstruct a complex project or problem into smaller discrete modules that can be developed independently, then incorporated together at a later time.

KS3.2.2 Deconstruct a complex problem into simpler parts using predefined constructs (e.g., functions and parameters and/or classes). 3A-A-4-8

O3.3 Explain how people participate in a problem-solving process that scales. LO 7.1.2 [P4]

KS3.3.1 Describe how human capabilities are enhanced by digitally enabled collaboration. EK 7.1.2D

KS3.3.2 Explain how services use the contributions of many people to benefit both individuals and society. EK 7.1.2E

KS3.3.3 Explain how the move from desktop computers to a proliferation of always-on mobile computers is leading to new applications. EK 7.1.2G

D4 Computational Tools and Techniques

Computing involves the application of collaboration tools, programming tools, mathematical principles, and techniques to manage developments.

O4.1 Select and apply appropriate computational tools and techniques to solve a problem or create value for others.

KS4.1.1 Select tools for collaborating for data collection, writing, or programming.

KS4.1.2 Gain understanding of software tools and services while creating computational artifacts. EK 1.2.1B

KS4.1.3 Apply computing tools and techniques to create computational artifacts including, but not limited to, programming integrated development environments (IDEs). EK 1.2.1C

KS4.1.4 Navigate and use unfamiliar documentation and public information to extend the student's own knowledge of a programming language or to achieve a computational approach to solve a problem.

O4.2 Apply a system of version control effectively.

KS4.2.1 Maintain successive versions of a digital product during development.

C2 Technical Knowledge and Skills

Every career field requires technical literacy and career-specific knowledge and skills to support professional practice.

D5 Data

Data and information facilitate the creation of knowledge. Managing and interpreting an overwhelming amount of raw data is part of the foundation of our information society and economy.

O5.1 Collect, organize, and explore real and simulated data.

KS5.1.1 Identify tools and creative methods to collect and process data.

D6 Algorithms

Algorithms are used to develop and express solutions to computational problems. Algorithms are fundamental to even the most basic everyday task.

O6.1 Develop an algorithm for implementation in a program. LO 4.1.1 [P2]

KS6.1.1 Understand that sequencing, selection, and iteration are building blocks of algorithms. EK 4.1.1A

KS6.1.2 Understand that sequencing is the application of each step of an algorithm in the order in which the statements are given. EK 4.1.1B

KS6.1.3 Use a Boolean condition or selection to determine which of two parts of an algorithm are used. EK 4.1.1C

KS6.1.4 Use Iteration or repetition of a part of an algorithm until a condition is met or until a specified number of times have been completed. EK 4.1.1D

KS6.1.5 (2c IWR) Combine algorithms to make new algorithms and explain how they function both independently and together. EK 4.1.1E

KS6.1.6 Use existing correct algorithms as building blocks for constructing a new algorithm to help ensure the new algorithm is correct. EK 4.1.1F

KS6.1.7 Identify different algorithms that can be developed to solve the same problem. EK 4.1.1H

KS6.1.8 Implement and analyze common patterns employing variables and iteration, including “for” loops iterating across a list, value accumulation, and list aggregation.

O6.2 Express an algorithm in a language. LO 4.1.2 [P5]

KS6.2.1 Contrast the languages for algorithms including natural language, pseudocode, and visual and textual programming languages. EK 4.1.2A

KS6.2.2 Write in natural language and pseudocode to describe algorithms so that humans can understand them. EK 4.1.2B

KS6.2.4 Understand that every algorithm can be constructed using only sequencing, selection, and iteration. EK 4.1.2G

D7 Abstraction

Abstraction reduces information and detail to facilitate focus on relevant concepts. It is a process, a strategy, and the result of reducing detail to focus on concepts relevant to understanding and solving problems.

O7.1 Describe the variety of abstractions used to represent data. LO 2.1.1 [P3]

KS7.1.1 Understand that high-level programming languages provide more abstractions for the programmer and make it easier for people to read and write a program. EK 2.2.3B

KS7.1.2 Identify what has been made more general by an abstraction and identify what details have been hidden or removed. EK6.1.2

KS7.1.3 Describe the role of abstraction in handling complexity. (e.g., abstraction in programming languages, procedural abstraction) EK6.1.1.

KS7.1.4 Identify advantages and disadvantages of working at high and low levels of abstraction.

O7.2 Describe an abstraction used when writing a program or creating other computational artifacts. LO 2.2.1 [P2]

KS7.2.1 Create an abstraction that generalizes functionality with input parameters that allow reuse. EK 2.2.1C

KS7.2.2 Represent multiple levels of abstractions, such as constants, expressions, statements, procedures, and libraries. EK 2.2.2A

KS7.2.3 (2d IWR) Describe how an abstraction is used to manage complexity in a specific program.

D8 Programming

Programming enables problem solving, human expression, and creation of knowledge. Any particular programming language is selected based on appropriateness for a specific project or problem.

O8.1 (2a IWR) Creative Expression in Programming - Develop a program for creative expression, to satisfy personal curiosity, or to create new knowledge. LO 5.1.1 [P2] Or (2b IWR)Problem Solving in Programming - Develop a program to solve problems. LO 5.1.

KS8.1.1 Develop programs used in a variety of ways by a wide range of people. EK 5.1.1A

KS8.1.2 Understand that programs developed for creative expression, to satisfy personal curiosity, or to create new knowledge may have visual, audible, or tactile inputs and outputs. EK 5.1.1B

O8.2 Iteration in Programming - Create programs by writing and testing code in a modular, incremental approach.

KS8.2.1 (2b IWR) Describe how an iterative process of program development helps in developing a correct program to solve problems. EK 5.1.2A

KS8.2.2 Incrementally add tested program segments to correct working programs to help create larger correct programs. EK 5.1.2C

KS8.2.3 Adapt or improve existing code.

O8.3 Algorithms in Programs - Explain how programs implement algorithms. LO 5.2.1 [P3]

KS8.3.1 Understand that program instructions are executed sequentially. EK 5.2.1B (A112)

KS8.3.2 Describe how program instructions may involve variables that are initialized and updated, read, and written. EK 5.2.1C

O8.4 Abstraction in Programs - (2d IWR) Use an abstraction to manage complexity in programs. LO 5.3.1 [P3]

KS8.4.1 Understand that procedures are reusable programming abstractions. EK 5.3.1A

KS8.4.2 Understand that a procedure is a named grouping of programming instructions. EK 5.3.1B

KS8.4.3 Use procedures to reduce the complexity of writing and maintaining programs. EK 5.3.1C

KS8.4.4 Understand that procedures have names and may have parameters and return values. EK 5.3.1D

KS8.4.6 Use parameters to generalize a solution by allowing a procedure to be used instead of duplicated code. EK 5.3.1F

KS8.4.7 Understand that parameters provide different values as input to procedures when they are called in a program. EK 5.3.1G

KS8.4.8 Use lists and list operations, such as add, remove, and search. EK 5.3.1K

O8.5 Mathematical and Logic Concepts in Programming - (2c IWR) Employ and describe appropriate mathematical and logical concepts in programming. LO 5.5.1 [P1]

KS8.5.1 Understand that strings and string operations, including concatenation and some form of substring, are common in many programs. EK 5.3.1

KS8.5.2 Recognize that numbers and numerical concepts are fundamental to programming. EK 5.5.1A

KS8.5.3 Understand that real numbers are approximated by floating-point representations that do not necessarily have infinite precision. EK 5.5.1C

KS8.5.4 Understand that mathematical expressions using arithmetic operators are part of most programming languages. EK 5.5.1D

KS8.5.5 Understand that logical concepts and Boolean algebra are fundamental to programming. EK 5.5.1E

KS8.5.6 Use compound expressions using and, or, and not. EK 5.5.1F

KS8.5.7 Use computational methods such as lists and collections to solve problems. EK 5.5.1H

KS8.5.8 Understand that lists and other collections can be treated as abstract data types (ADTs) in developing programs. EK 5.5.1I

KS8.5.9 Understand there are basic operations on collections including adding elements, removing elements, iterating over all elements, and determining whether an element is in a collection. EK 5.5.1J

D9 Modeling and Simulation

People use computer programs to process information to gain insight and knowledge.

O9.1 Use models and simulations to represent phenomena. LO 2.3.1 [P3]

KS9.1.1 Understand that models and simulations are simplified representations of more complex objects or phenomena. EK 2.3.1A

KS9.1.2 Understand that models may use different abstractions or levels of abstraction depending on the objects or phenomena being posed. EK 2.3.1B

KS9.1.3 Understand that simulations mimic real-world events without the cost or danger of building and testing the phenomena in the real world. EK 2.3.1D

D10 The Internet

The internet pervades modern computing. The internet and the systems built on it have had a profound impact on society. Computer networks support communication and collaboration.

O10.1 Explain characteristics of the internet and the systems built on it. LO 6.2.1 [P5]

KS10.1.1 Understand that interfaces and protocols enable widespread use of the internet. EK 6.2.2D

KS10.1.2 Use application programing interfaces (APIs) to connect all devices.

D11 Cybersecurity

Cybersecurity is an important concern for the internet and the systems built on it.

O11.1 Identify existing cybersecurity concerns and potential options to address these issues with the internet and the systems built on it. LO 6.3.1 [P1]

KS11.1.1 Explain how implementing cybersecurity has software, hardware, and human components. EK 6.3.1C

KS11.1.2 Understand that cyberwarfare and cybercrime have widespread and potentially devastating effects. EK 6.3.1D

KS11.1.3 Understand that phishing, viruses, and other attacks have human and software components. EK 6.3.1F

O11.2 Identify user actions that strengthen the security of a networked computing system.

KS11.2.1 Describe secure practices related to passwords, antivirus software, software updates, and posting content online.

KS11.2.2 Identify the unique circumstances in which penetration testing is legal and ethical.

C3 Professional Practices and Communication

Professional practice is guided by professional ethics and standards and requires effective communication and collaboration.

D12 Social Impacts of Computing

Cybersecurity affects economic, environmental, and societal contexts.

O12.1 Explain the connections between computing and real-world contexts, including economic, social, and cultural contexts. LO 7.4.1

KS12.1.1 Describe how mobile, wireless, and networked computing has an impact on innovation throughout the world. EK 7.4.1B

D13 Career Awareness

Today computing impacts almost all careers. There are career specializations within computer science such as software development, security, network, and systems administration.

O13.1 Describe career paths within the computing specialties.

KS13.1.1 Describe a variety of careers within the computing specialties.

KS13.1.2 Recognize the education and credentialing requirements for careers within computing specialties.

KS13.1.3 Demonstrate the initiative and independent learning required to stay current with evolving technology and career needs.

O13.2 Explain how computing has impacted innovation in other fields. LO 7.2.1 [P1]

KS13.2.1 Describe how scientific computing has enabled innovation in science and business. EK 7.2.1B

KS13.2.2 Describe how computing enables innovation by providing the ability to access and share information. EK 7.2.1C

KS13.2.3 Describe how advances in computing has enabled technology to generate and increase the creativity in other fields. EK 7.2.1G

D14 Professionalism and Ethics

Computing professionals must make decisions regularly regarding their professional and social conduct when collaborating with developers and engaging with users to get feedback.

O14.1 Abide by professional standards when creating value for people and society.

KS14.1.1 Create and maintain a secure professional identity for accessing IDEs and accessing computer science communities.

KS14.1.2 Provide rationales for all ethical decisions.

KS14.1.3 Engage others with respect and forethought.

O14.2 Access, manage, and attribute information using effective strategies. LO 7.5.1 [P1]

KS14.2.1 Understand that online databases and libraries catalog and house secondary and some primary sources. EK 7.5.1A

KS14.2.2 Understand that plagiarism is a serious offense that occurs when a person presents another’s ideas or words as his or her own. Plagiarism may be avoided by accurately acknowledging sources. EK 7.5.1C

O14.3 Consider accessibility and equity when designing products, creating solutions, and collaborating with others.

KS14.3.1 Explain how diversity on development teams is essential for producing outcomes that serve a diverse audience.

O14.4 Evaluate online and print sources for appropriateness and credibility. LO 7.5.2 [P5]

KS14.4.1 Evaluate the credibility of a source by considering reputation and credentials of the author(s), publisher(s), site owner(s), and/or sponsor(s). EK 7.5.2A

KS14.4.2 Evaluate the relevancy of information from a source and if it supports an appropriate claim or the purpose of the investigation. EK 7.5.2B

D15 Collaboration

Diverse perspectives, good interpersonal relationships, and effective collaboration strategies generate the most robust and innovative solutions.

O15.1 Collaborate when processing information to gain insight and knowledge. LO 3.1.2 [P6]

KS15.1.1 Understand that collaboration is an important part of solving data-driven problems. EK 3.1.2A

KS15.1.2 Understand that collaboration facilitates solving computational problems by applying multiple perspectives, experiences, and skill sets. EK 3.1.2B

KS15.1.3 Understand that communication between participants working on data-driven problems gives rise to enhanced insights and knowledge. EK 3.1.2C

KS15.1.4 Understand that collaborating face-to-face and using online collaborative tools can facilitate processing information to gain insight and knowledge. EK 3.1.2E

O15.2 Collaborate to develop a program. LO 5.1.3 [P6]

KS15.2.1 Understand that collaboration can decrease the size and complexity of tasks required of individual programmers. EK 5.1.3A

KS15.2.2 Understand that collaboration facilitates multiple perspectives in developing ideas for solving problems by programming. EK 5.1.3B

KS15.2.3 Understand that collaboration in the iterative development of a program requires different skills than developing a program alone. EK 5.1.3C

KS15.2.4 Understand that collaboration can make it easier to find and correct errors when developing programs. EK 5.1.3D

KS15.2.5 Understand that collaboration facilitates developing program components independently. EK 5.1.3E

O15.3 Apply project management strategies effectively as part of a team.

KS15.3.1 Prioritize short-term and long-term objectives using an Agile methodology when working on a project.

KS15.3.2 Select and use computational tools that enable collaboration.

KS15.3.3 Work with a group to establish team norms.

KS15.3.4 Establish clear responsibilities and split workloads equitably.

D16 Communication

Computing professionals must be able to explain and justify the design and appropriateness of their computational choices, and analyze and describe both computational artifacts and the results or behaviors of such artifacts.

O16.1 Communicate ideas, processes, and products to optimize audience perception and understanding.

KS16.1.1 Create program documentation that helps programmers develop and maintain correct programs to efficiently solve problems. EK 5.1.2D

KS16.1.2 Create documentation about program components, such as code segments and procedures, that helps in developing and maintaining programs. EK 5.1.2E

KS16.1.3 Create documentation that helps in developing and maintaining programs when working individually or in collaborative programming environments. EK 5.1.2F

KS16.1.4 Summarize the purpose of a computational artifact.

KS16.1.5 (2b IWR) Communicate which portions of a program you developed independently and which were created collaboratively.