# Motivation and Principles behind Essential Computing Concepts

The development of CS2023 invited two years' worth of conversations amongst CS educators around the world about what the essential elements are of a computer science undergraduate curriculum. These conversations re-surfaced a long known tension between the merits of curriculum guidelines that are sufficiently detailed to articulate a full and sufficient set of content to define the required elements of a CS major, and the merits of smaller guidelines that leave greater space for interpretation and institution-specific customization. While schools of any type may prefer the latter format, smaller institutions, liberal arts colleges, and computing programs looking to innovate around the combination of CS with other disciplines have a particular desire for a more minimal articulation of the essential core of a computer science or computing major.

The Essential Computing Concepts Working Group[1] of the SIGCSE Committee on Computing Education in Liberal Arts Colleges was formed in Spring 2025 following two sessions[2] at SIGCSE 2025 that were used to solicit community input on the desire for alternative guidelines. This group met monthly for the remainder of the calendar year, as well as in smaller subgroups between meetings, to study this community input and develop a proposed set of guidelines.

The Essential Computing Concepts Working Group membership included:

Rick Blumenthal, Regis University
Alyce Brady, Kalamazoo College
Grant Braught, Dickinson College
Janet Davis, Whitman College
Amanda Holland-Minkley, Washington & Jefferson College
Bruce Maxwell, Colby College
Megan Olsen, Loyola University Maryland
Karl Schmitt, Trinity Christian College
Andrea Tartaro, Furman University
James Teresco, Siena University
Henry Walker, Grinnell College

The working group members appreciate the many conversations they have had with their institutional colleagues that informed their contributions to this project.

[1] This group was originally called the Microkernel Working Group based on early terminology around defining a smaller set of guidelines as a "microkernel" of the larger CS2023 core. However, as the group's work progressed and began to deviate not just in the size of the guidelines but also their structure, the group determined the name "Essential Computing Concepts" would be more appropriate.

[2] https://dl.acm.org/doi/10.1145/3641555.3705098 and https://computing-in-the-liberal-arts.github.io/SIGCSE2025-Affiliated-Event/

## Liberal Arts Context

Early on, the working group considered a number of different goals that a smaller set of guidelines might achieve. Based on the group's composition and sponsorship, we were committed to developing a smaller core that would be useful to liberal arts institutions, with their values of flexibility, exploration, and interdisciplinary work within a CS curriculum. However, we attempted not to overly-specialize to a liberal arts context and design guidelines that would be useful to any institution that would find a smaller core useful.

That being said, a liberal arts philosophy did guide our work. The SIGCSE-LACS Committee's working definition of liberal arts education refers to "a philosophy of higher education that emphasizes preparing students for the full range of thinking they will face throughout their lives: thinking in the service of a career, thinking in order to participate in civic affairs and society generally, thinking in order to have a fulfilling live, etc." "Liberal arts computing education thus covers any computing courses or curricula designed with the goal of educating students broadly for life in addition to any professional or career goals that are also supported." [https://dl.acm.org/doi/10.1145/3314027]

Broadly speaking, common themes of the vision for liberal arts education consider the multiple goals of an education and a prioritization of critical thinking and skill development that apply across one's life rather than in a specific career. Connections between disciplines and between academic and experiential learning are emphasized, in contrast to learning outcomes that are siloed to particular disciplines or courses of study. This is consistent with the focus of computing in liberal arts on interdisciplinarity, computer science as an approach to reasoning and problem-solving, and flexibility to support a variety of career paths.

Thus, while we hope that many types of institutions will find the Essential Computing Concepts useful, we have particularly focused on developing guidelines that will be useful to CS programs that consider it their educational mission to:

- Develop skills in the reasoning, representation, abstraction, and analysis techniques that characterize computational thinking and problem-solving in computer science (a "computational mindset"), independent of where they may be applied;
- Emphasize connections between disciplines, including both the ways that the discipline of computing can utilize the content and skills of other disciplines, and the ways that other disciplines can utilize computational perspectives and techniques;
- Emphasize those elements of computer science that further education for citizenship and life, as well as for a career.

## Goals for the Project

Given this context, the group asked what properties would be needed for a defined set of essential computing concepts to be useful for this audience. Some structural considerations were evident, such as being consistent with what can feasibly be taught within a small program with limited faculty. It should be small enough in scope to ensure space for academic exploration

or other student priorities. We also wanted the set to be suitable as the core of a CS major or the basis for a CS minor or CS+X program.

From a design perspective, we wanted any definition of essential computing concepts to achieve the following:

- Support novel curricular structures and be formulated not to assume any pre-requisite content or skills
- Can be spread across a selection of courses in multiple ways, such as through a spiral pedagogical pattern
- Capture a set of necessary principles or concepts for a CS program to cover without attempting to define what is sufficient for a CS major
- Prioritize principles or concepts that provide grounding for higher-level ideas in multiple CS sub-areas and which would need to be covered in multiple different upper-level courses if omitted from introductory courses
- Emphasize concepts, skills, and techniques applicable across a broad range of contexts over those applicable to limited/specialized contexts and support a range of next steps students might pursue in computing
- Sufficient breadth will be required of students to ensure they have practiced a computational mindset in enough different contexts to be able to transfer the mindset and apply it to novel problems, though the selection of that breadth need not be fixed
- Include concepts, skills, and techniques that can be taught in an interdisciplinary manner
- Include concepts, skills, and techniques that can be taught in a collaborative or project-based manner
- Balance the needs of a narrow career focus in technology, a wide career focus for any professional trajectory, and a focus on lifelong learning
- Be forward thinking about what CS knowledge will be important in the age of generative AI and will remain stable as the core of the discipline even as new technologies emerge
- Capture the “big, beautiful, enduring ideas” of computer science

During this process, the group also benefited from Henry Walker’s work contrasting the approaches taken for expressing curricular guidelines by ACM/IEEE versus the Mathematical Association of America (MAA). The MAA guidelines provided a model of a smaller set of guidelines that focus on high-level curricular goals and permit departments to determine how to select specific content or specialization with which to achieve those goals. We encourage people to read more about his exploration of these two curricular approaches in his January 2026 Inroads article. [https://dl.acm.org/doi/10.1145/3779069]

## Design of the Essential Computing Concepts

Based on these design principles, several members of the working group went through brainstorming exercises considering what content might be essential for all students. In reviewing these lists, we were able to identify high-level learning outcomes that a set of essential computing concepts should support. These learning outcomes were informed by our liberal arts perspective and the variety of student populations that they might support. While a student might best achieve these learning outcomes after completing a CS or computing major,

these outcomes (or a subset of them) could also be achieved within a minor or smaller collection of courses.

These high-level learning outcomes are as follows:

- Students will be able to produce artifacts that run on a computer that are appropriately designed and constructed to address a given challenge.
- Students will be able to evaluate and test designs and solutions to problems, whether generated by themselves or others.
- Students will be able to think across levels of abstraction. Given a problem, they should be able to have multiple mental models at multiple levels of abstraction.
- Students will be able to collaborate and communicate with others in a manner that supports group-based computational understanding and problem-solving.
- Students will be able to identify how a problem can be approached from a computer science perspective, though they may need to learn more or consult an expert to actually solve the problem.
- Given a multifaceted or wicked problem, students will be able to identify the role computer scientists can play in responsibly working to address that problem.
- Students will be prepared to use the foundational content of computer science covered within the essential computing concepts to learn that computer science content which is not covered.

Our decision to focus on learning outcomes rather than content is consistent with our intent to provide guidelines that provide programs with flexibility in implementation. Based on these high-level outcomes, we proceeded with developing a collection of more specific outcomes that would be more useful in guiding the design or review of a curriculum or set of courses. These outcomes are grouped into four major types of tasks that students ought to be able to achieve: Read, Understand, and Analyze Models; Design and Create Using Models; Use Math, Data and Analytics for Computer Science; and Be a Responsible Computing Practitioner. These learning outcomes, with lists of essential concepts and illustrative examples, are available here: https://tinyurl.com/ECCSIGCSE