Introduction to Knowledge Model

Definitions and Terminology

A **knowledge model** of a curriculum is structured as a set of knowledge areas:

$$\text{Knowledge model} = \{ \text{Knowledge areas} \}$$

A **knowledge area** is a set of related knowledge units:

$$\text{Knowledge area} = \{ \text{Knowledge units} \}$$

A **knowledge unit** is a set of related topics and a set of learning outcomes for those topics:

$$\text{Knowledge unit} = \{ \text{Topics} \} + \{ \text{Learning outcomes} \}$$

Learning outcomes are used for assessment. Each topic in a knowledge unit is categorized as either core or elective:

$$\text{Topic} \in \{ \text{core} \} \cup \{ \text{elective} \}$$

Core topics are topics that *every* graduate **must** know. Every curriculum is typically expected to cover all the core topics.

Elective topics are those that are not required of every graduate. Nevertheless, they complement the coverage of core topics. In addition to covering all the core topics, a curriculum is expected to cover a considerable percentage of elective topics.

Students may be expected to demonstrate proficiency in topics at different **skill levels**. Typically, Bloom's taxonomy [8] is used to describe skill levels. The instructional time needed to cover a topic is determined by the skill level, e.g., instructing how to apply a concept may take longer than instructing how to explain the concept.

The expected skill levels and, therefore, the time needed to cover topics are salient because they determine how many topics can be packaged into a typical **course** and how many courses are needed in a **curriculum** to cover all the required topics. Thus, the list of core topics and the skill levels at which students must demonstrate proficiency in those topics determine the minimum size of a curriculum.

CS2023 Knowledge Model

Knowledge areas: The CS2023 knowledge model consists of 17 knowledge areas, listed in alphabetical order of their abbreviation:

● Artificial Intelligence (AI) ● Algorithmic Foundations (AL) ● Architecture and Organization (AR) ● Data Management (DM) ● Foundations of Programming Languages (FPL) ● Graphics and Interactive Techniques (GIT) ● Human-Computer Interaction (HCI) ● Mathematical and Statistical Foundations (MSF)	● Networking and Communication (NC) ● Operating Systems (OS) ● Parallel and Distributed Computing (PDC) ● Software Development Fundamentals (SDF) ● Software Engineering (SE) ● Security (SEC) ● Society, Ethics, and the Profession (SEP) ● Systems Fundamentals (SF) ● Specialized Platform Development (SPD)

Details of the knowledge areas are in Section 3 of this report. In CS2023, knowledge areas have been expanded to include professional dispositions in addition to knowledge units:

Knowledge area = { Knowledge units } + { Professional dispositions }

Professional dispositions are malleable values, beliefs, and attitudes that enable consistent behaviors desirable in the workplace, e.g., *persistent*, *self-directed*. When dispositions seem indistinguishable from skills (e.g., *communicative*, *collaborative*), they refer to the willingness and intent to apply the skills to complete a task. They are sought by employers and are essential for succeeding in the workplace. Most recent curricular guidelines have emphasized the need for and the value of professional dispositions in computing education, including Information Technology 2017 [4], Computing Curricula 2020 [5], Information Systems 2020 [6], and Data Science 2021 [7]. These guidelines have proposed competency models of the curricula in which knowledge, skills, and professional dispositions needed to carry out a task are bundled together.

Dispositions vary by knowledge area. Some dispositions are more important at certain stages in a student's development than others, e.g., being *persistent* is essential at introductory levels, whereas being *self-directed* is expected at advanced levels of study. Group projects call for *collaborative* disposition whereas mathematical foundations demand *meticulous* disposition. So, associating dispositions with knowledge areas instead of individual tasks makes it easier for educators to repeatedly and consistently promote dispositions during the accomplishment of tasks relevant to the knowledge area. In CS2023, the professional dispositions most relevant for each knowledge area have been listed.

In CS2023, core topics are categorized as either CS Core or KA Core. Elective topics are renamed Non-core:

Topic $\in$ { CS Core } **U** { KA Core } **U** { Non-core }

CS (Computer Science) Core topics are topics that *every* computer science graduate **must** know. Every computer science curriculum is expected to cover all the CS Core topics. **KA (Knowledge Area) Core** topics are topics *recommended* for more in-depth study. Additional nuances to the concept of KA Core topics are:

- Topics in Mathematical and Statistical Foundations (MSF) knowledge area are designated as KA Core to indicate that they are required for KA Core topics in other knowledge areas, e.g., many KA Core topics in statistics are needed for KA Core topics in the Artificial Intelligence (AI) knowledge area.
- Some knowledge areas have more than one subset of KA Core topics, e.g., Specialized Platform Development (SPD) has KA Core topics on web platforms, mobile platforms, embedded platforms, etc. with minimal overlap among them. The hours for these disparate KA Core topics are not to be considered additive for the knowledge area.

Over 50% of the knowledge units in the 17 knowledge areas of CS2023 include CS Core topics and over 75% of the knowledge units include KA Core topics. A curriculum may choose to focus on the KA Core topics of some knowledge areas in greater depth/breadth than others. The subset of knowledge areas on which a curriculum is focused, when coherently chosen, defines the **competency area**(s) of the curriculum.

$$\text{Competency area} \subseteq \text{Knowledge model}$$

Three *representative* competency areas are presented in CS2023:

- **Software** Development – the knowledge areas that prepare a student to be a journeyman programmer. These include Software Development Fundamentals (SDF), Algorithmic Foundations (AL), Foundations of Programming Languages (FPL), and Software Engineering (SE) knowledge areas.
- **Systems** Development – the knowledge areas that prepare a student to provide essential services including non-functional requirements. These include Systems Fundamentals (SF), Architecture and Organization (AR), Operating Systems (OS), Parallel and Distributed Computing (PDC), Networking and Communication (NC), Security (SEC), and Data Management (DM).
- **Applications** Development – the knowledge areas that prepare a student with problem-specific or solution-specific knowledge in addition to software development. These include Graphics and Interactive Techniques (GIT), Artificial Intelligence (AI), Specialized Platform Development (SPD), Human-Computer Interaction (HCI), Security (SEC), and Data Management (DM).

Society, Ethics, and the Profession (SEP) and Mathematical and Statistical Foundations (MSF) are part of all competency areas. Note that the Software competency area is a prerequisite of the other two competency areas. This list of competency areas is meant to be neither prescriptive nor comprehensive. Other competency area(s) may be based on institutional mission and local needs. Examples include *Computing for the social good, Scientific computing,* and *Secure computing*.

Skill levels: Four **skill levels** were adopted for use in CS2023: Explain, Apply, Evaluate, and Develop. They are loosely aligned with the revised Bloom's taxonomy [8] as shown in Table 1. *Explain* is the prerequisite skill for the other three levels. The verbs corresponding to each of the four skill levels were adopted from the work of ACM and CCECC [9].

<table>
<tr><th>Revised Bloom's Taxonomy</th><th>Skill level with applicable verbs</th></tr>
<tr><td>Remember</td><td rowspan="2">Explain: define, describe, discuss, enumerate, express, identify, indicate, list, name, select, state, summarize, tabulate, translate</td></tr>
<tr><td>Understand</td></tr>
<tr><td>Apply</td><td rowspan="2">Apply: backup, calculate, compute, configure, debug, deploy, experiment, install, iterate, interpret, manipulate, map, measure, patch, predict, provision, randomize, recover, restore, schedule, solve, test, trace, train, virtualize</td></tr>
<tr><td rowspan="2">Analyze</td></tr>
<tr><td rowspan="2">Evaluate: analyze, compare, classify, contrast, distinguish, categorize, differentiate, discriminate, order, prioritize, criticize, support, decide, recommend, assess, choose, defend, predict, rank</td></tr>
<tr><td>Evaluate</td></tr>
<tr><td>Create</td><td>Develop: combine, compile, compose, construct, create, design, generalize, integrate, modify, organize, plan, produce, rearrange, rewrite, refactor, write</td></tr>
</table>

Table 1. Skill levels and corresponding verbs and levels in revised Bloom's taxonomy.

In CS2023, desired skill levels were identified for CS Core and KA Core topics. The desired skill levels were then used to estimate the time needed for the instruction of CS Core and KA Core topics. These details are presented in tabular format in Section 3 of this report. The skill levels identified for core topics should be treated as recommended, not prescriptive.

The time needed to cover CS Core and KA Core topics is expressed in terms of **instructional hours**. Instructional hours are hours spent in the classroom imparting knowledge regardless of the pedagogy used. Students are expected to spend additional time after class practicing related skills and exercising professional dispositions.

In CS2023, CS Core topics at the desired skill levels are estimated to need 270 hours of instructional time. Since every computer science graduate must know CS Core topics, every computer science curriculum is expected to include all 270 hours of instruction. While CS Core topics set the *minimum* for a computer science curriculum, a typical curriculum is expected to include many more KA Core topics in a competency area of choice as well as complementary Non-core topics.

A curriculum is structured in terms of courses, not knowledge areas. A knowledge area is not necessarily a course. Many courses may be carved out of a knowledge area and a course may contain topics from multiple knowledge areas. In CS2023, recommendations have been provided in each knowledge area for **packaging course(s)** from its topics. For packaging purposes, a course is assumed to meet for around 40 instructional hours.

The number of courses in a computer science curriculum varies among educational institutions. In acknowledgment of this variety, in CS2023, **curricular packaging** recommendations have been provided in Section 3 for programs that require 8, 10, 12, and 16 computer science courses. Both course and curricular packaging recommendations are suggestive, not prescriptive. It is expected that curriculum designers will adapt them to suit their local needs, resources, and constraints. The packaging recommendations may also be used to compare courses and curricula of different educational institutions.

Structure of CS2023 Knowledge Area Explained

Each of the 17 knowledge areas in Section 3 is structured as follows.

- **A Preamble** that describes the knowledge area and includes:
  - Changes in the knowledge area since CS2013;
  - An optional overview of the various knowledge units in the knowledge area.
- A table listing CS Core and KA Core hours assigned to the knowledge area.
- A listing of knowledge units, each of which in turn consists of:
  - CS Core, KA Core and Non-core topics, enumerated, and
  - CS Core and KA Core illustrative learning outcomes, enumerated.
- Professional dispositions most relevant to the knowledge area.
- Mathematics requirements for the knowledge area—both required and desired.
- Suggestions for packaging courses from the knowledge area.
- The committee members who reviewed and revised the knowledge area recommendations.

Table of Core Hours: The table lists the number of CS Core and KA Core hours assigned to each knowledge unit in the knowledge area. Where applicable, the table also lists the CS/KA Core hours shared with other knowledge areas.

Knowledge units: Each knowledge unit has an abbreviated name in addition to its complete name. For example, the abbreviated name of the knowledge unit “Computational Models and Formal Languages” in Algorithmic Foundations (AL) is “Models.” The abbreviated name is used to refer to the knowledge unit in the rest of the document, as in “AL-Models.”

Topics: Within each knowledge unit, topics are categorized as CS Core, KA Core or Non-core. They are consecutively numbered through all three categories so that each topic can be uniquely identified. The enumeration should not be construed as implying any order or dependency among the topics. The recommended syntax for referring to a topic is:

<Knowledge area abbreviation>--<Knowledge unit abbreviation>: Decimal.alphabetic.roman

For example, “Local vs global solutions” in the following is **AI-Search: 3.c.i**:

3. Heuristic graph search for problem-solving
    c. Local minima and the search landscape
        i. Local vs global solutions

Even though knowledge areas are groupings of related topics, they are not insular. Making connections among the various knowledge areas is an essential part of maturing as a computer science graduate. So, whenever possible, relationships between topics in different knowledge areas have been pointed out with “See also:” annotation in CS2023. For example, for the topic “Attacks and antagonism” in OS-Protection, the reader is directed to see also SEC-Foundations.

Typically, when a topic appears in two knowledge areas, the two knowledge areas present different perspectives on the topic. When one knowledge area is the primary repository for a topic and the other knowledge area refers to the topic from the repository, “See also” annotations are skipped in the knowledge area that serves as the primary repository. The knowledge areas that serve as primary repositories include Software Development Fundamentals (SDF – all the introductory programming topics), Algorithmic Foundations (AL – all the algorithms), Mathematical and Statistical Foundations (MSF), and Society, Ethics, and the Profession (SEP).

Learning Outcomes: In each knowledge unit, learning outcomes have been identified primarily for CS Core and KA Core topics. Furthermore, they have been listed under CS Core and KA Core subtitles and have been consecutively numbered for ease of referencing. The learning outcomes are meant to be descriptive, not prescriptive. So, they have been re-named **Illustrative Learning Outcomes**.

Professional Dispositions: The professional dispositions most relevant to each knowledge area have been listed, along with a brief explanation of why they are relevant to the knowledge area. These dispositions are desirable for most of the tasks that require a knowledge of this knowledge area.

The list of dispositions is not meant to be comprehensive, but rather, to provide a starting point for adaptation. Dispositions, by definition, vary with the context—type of institution, academic level of students, demographic profile of the student body, etc. Educators will want to adapt this list of dispositions to suit their local context.

Dispositions are not meant to be used to exclude students based on their background or demographics. Instead, they are listed in the spirit that explicit acknowledgment of their role will provide impetus for educators to foster them in their curricula and level the playing field for the success of all computer science graduates regardless of their background or prior preparation.

Mathematics requirements: The mathematics requirements have been individually noted for each knowledge area. Where applicable, the requirements have been further categorized as either required or desirable. This serves several purposes: 1) it acknowledges the essential role played by mathematics for success in computer science; 2) it provides for the possibility of covering the mathematics required for a computer science course within the course instead of as a prerequisite gatekeeper mathematics course; and 3) it helps educators provide access ramps for computer science students underprepared in mathematics.

Course packaging suggestions: Each course has been specified in terms of knowledge units, both from within and outside the knowledge area. Instructional hours have been suggested for each knowledge unit. These hours represent the weight suggested for the knowledge unit in a

typical course of around 40 instructional hours. A course that meets for fewer or more hours may scale the hours accordingly and/or include fewer/more knowledge units in its coverage, as long as the course covers all the listed CS Core topics. Finally, objectives have been specified for each course that describe what a student must be able to do once the student has completed the course.

Committee: The committee that reviewed and revised the knowledge area has been listed along with its chair. The committee has typically included international experts from academia and industry who met regularly over the course of the curricular revision. In addition, non-committee experts who contributed significantly to the knowledge area have been listed as Contributors. Experts who reviewed the drafts of the knowledge area have been listed as Reviewers in the Acknowledgments section.

Changes since CS2013

The CS2023 knowledge model was developed by revising the CS2013 curricular guidelines [3]. Significant changes from CS2013 to CS2023 are described below in terms of the components of a knowledge model.

Knowledge Areas: Several CS2013 knowledge areas were renamed, either to better emphasize their focus or to incorporate changes in the area since 2013.

- Algorithms and Complexity (AL) as Algorithmic Foundations (AL)
- Discrete Structures (DS) as Mathematical and Statistical Foundations (MSF)
- Graphics and Visualization (GV) as Graphics and Interactive Techniques (GIT)
- Information Assurance and Security (IAS) as Security (SEC)
- Information Management (IM) as Data Management (DM)
- Intelligent Systems (IS) as Artificial Intelligence (AI)
- Platform Based Development (PBD) as Specialized Platform Development (SPD)
- Programming Languages (PL) as Foundations of Programming Languages (FPL)
- Social Issues and Professional Practice (SP) as Society, Ethics, and the Profession (SEP)

Computational Science (CN) from CS2013 was dropped as a knowledge area because it had very little computer science content that was not also included in other knowledge areas. In CS2013, it was allocated just one core hour for modeling and simulation. Modeling was considered and rejected as an alternative to Computational Science: while applying modeling is a crosscutting theme in computer science, studying modeling for its own sake more appropriately belongs in the CS + X space.

Given the increased role played by mathematics in computer science today, the mathematical component of computer science was expanded from Discrete Structures in CS2013 to also include probability, statistics, and linear algebra. The expanded knowledge area was renamed Mathematical and Statistical Foundations (MSF) to reflect this change.

Systems Fundamentals (SF) from CS2013 was considered for elimination, but ultimately retained since it provides a system-wide perspective not also available in any of the other

systems knowledge areas, that is, Architecture and Organization (AR), Operating Systems (OS), Networking and Communication (NC) or Parallel and Distributed Computing (PDC).

Data Science was considered for inclusion as a new knowledge area in CS2023, but ultimately rejected since all of Data Science's computer science-specific topics are already covered by Artificial Intelligence (AI), Graphics and Interactive Techniques (GIT), Data Management (DM) and Mathematical and Statistical Foundations (MSF).

SEP Knowledge Unit: Given that the work of computer science graduates affects all aspects of everyday life, computer science as a discipline can no longer ignore or treat as incidental, social, ethical, and professional issues. In recognition of this pervasive nature and influence of computing, effort was made to include a separate knowledge unit on Society, Ethics, and the Profession (SEP) in every other knowledge area. Topics and learning outcomes at the intersection of the knowledge area and SEP were explicitly listed in the knowledge unit to help educators call attention to these issues across the curriculum.

Topics: In CS2023, as stated earlier, every topic has been provided a unique identifier so that any topic can be unambiguously referenced using the notation:

<Knowledge area abbreviation>--<Knowledge unit abbreviation>: Decimal.alphabetic.roman

Skill Levels: The three skill levels used in CS2013 have been expanded to four. The skill levels used in CS2013 were: Familiarity ("What do you know about this?"), Usage ("What do you know how to do?") and Assessment ("Why would you do that?"). In CS2023, in order to reflect the learner's thinking processes and actions rather than behaviors, verbs were used for skill levels instead of nouns: Familiarity was renamed Explain; and Assessment was renamed Evaluate. Usage was split into two: Apply and Develop. Apply was introduced as a skill level because of the increased emphasis placed on it in online courseware. It is also a skill level of increasing importance in light of the availability of generative AI for development tasks. The four skill levels, that is, Explain, Apply, Develop, and Evaluate, are loosely aligned with the revised Bloom's taxonomy [8] as shown earlier in Table 1.

Learning Outcomes: In CS2013, each learning outcome was labeled with a skill level. Since the skill level of a learning outcome can typically be inferred from the verb used in the learning outcome statement, in CS2023, skill level labels were dropped from learning outcomes. Since the learning outcomes in CS2023 are meant to be descriptive, not prescriptive, they have been renamed *Illustrative Learning Outcomes* to acknowledge that additional learning outcomes can be specified at other skill levels for each topic.

Professional Dispositions: CS2013 guidelines [3] emphasized the importance of dispositions in passing (Professional Practice, pp. 15-16). In addition to the knowledge model of CS2013, a framework for a competency model was also attempted in CS2023. A competency model demands a more integrated treatment of dispositions. So, in CS2023, professional dispositions appropriate for each knowledge area were identified.

Core Topics: In CS2013 [3], core topics were identified at two levels: Tier 1 accounting for 165 instructional hours and Tier 2 accounting for 143 hours. Computer science programs were expected to cover 100% of Tier 1 core topics and at least 80% of Tier 2 topics as shown in

Figure 1. While proposing this scheme, CS2013 was mindful that the number of core hours has been steadily increasing in curricular recommendations, from 280 hours in CC2001 [1] to 290 hours in CS2008 [2] and 308 hours in CS2013 [3]. Accommodating the increasing number of core hours poses a challenge for computer science programs that may want to restrict the size of the program either by design or due to necessity.

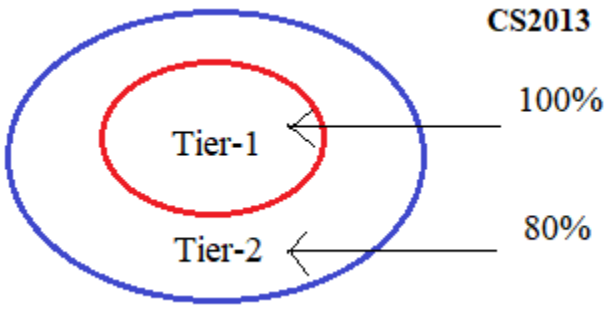

Figure 1: CS2013 Core Topics

In CS2023, instead of Tier 1 and Tier 2, core topics were categorized as CS Core and KA Core: every computer science program must cover all the CS Core topics. But a program may choose to cover KA Core topics in great detail in some knowledge areas and minimally or not at all in other knowledge areas as illustrated by highlighting in Figure 2. The result is a sunflower model that acknowledges that often, the design of curricula in smaller programs is dictated by curricular emphasis based on regional needs, the local availability of instructional expertise, and evolutionary history of the programs.

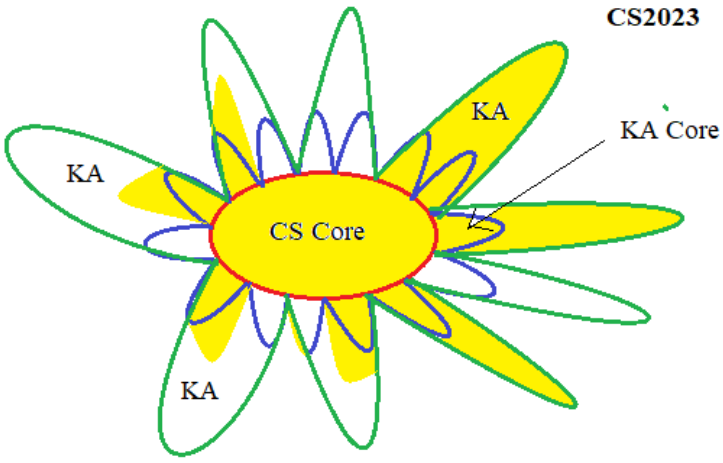

Figure 2: Sunflower model of core topics used in CS2023

Core hours: Table 2 shows how the distribution of core hours among the knowledge areas has changed from CS2013 to CS2023.

Knowledge Area	CS2013		CS2023	
	Tier-1	Tier-2	CS Core	KA Core
Artificial Intelligence (AI)	0	10	12	18
Algorithmic Foundations (AL)	19	9	32	32
Architecture and Organization (AR)	0	16	9	16
Data Management (DM)	1	9	10	26
Foundations of Programming Languages (FPL)	8	20	21	19
Graphics and Interactive Techniques (GIT)	2	1	4	70
Human-Computer Interaction (HCI)	4	4	8	16
Mathematical and Statistical Foundations (MSF)	37	4	55	145
Networking and Communication (NC)	3	7	7	24
Operating Systems (OS)	4	11	8	13
Parallel and Distributed Computing (PDC)	5	10	9	26
Software Development Fundamentals (SDF)	43	0	43	
Software Engineering (SE)	6	22	6	21
Security (SEC)	3	6	6	35
Society, Ethics, and the Profession (SEP)	11	5	18	14
Systems Fundamentals (SF)	18	9	18	8
Specialized Platform Development (SPD)	0	0	4	
Computational Science (CN)	1	0	Dropped	
Total	**165**	**143**	**270**	**N/A**

Table 2. Change in the distribution of core hours from CS2013 to CS2023.

Even though most knowledge areas in CS2023 contain a knowledge unit on Society, Ethics, and the Profession (SEP), core topics and hours for SEP issues were identified only in the Society, Ethics, and the Profession (SEP) knowledge area and not in the SEP knowledge units of other knowledge areas. This omission is meant to give educators the flexibility to decide how to distribute the coverage of the core SEP topics across the various computer science courses in their curriculum.

Course and Curricular Packaging: In CS2013, course and curricular exemplars were included from various institutions. But exemplars encapsulate institutional context such as the level of preparedness of students, the availability of teaching expertise, the availability of pre- and co-requisite courses, etc. that hinders their adoption across institutions. So, in CS2023, canonical packaging of courses has been provided in terms of knowledge areas and knowledge units as was done in CC2001 [1].

Designing/Revising a Curriculum Based on the Knowledge Model

A process for designing or revising a computer science curriculum using the CS2023 knowledge model is as follows.

1. Identify the **competency area(s)** to be targeted by the curriculum based on local needs. Some representative competency areas are Software, Systems, and Applications.

2. Based on the competency area(s) identified in step 1, select the **knowledge areas** whose **KA Core** topics will be covered in greater depth, while considering the availability of local resources (instruction, laboratory, etc.).
3. For each knowledge area identified in step 2, start with one or more **course packaging** suggestions. For each course add/subtract/scale **knowledge units** as appropriate.
4. Within each knowledge unit identified in step 3:
    a. Ensure that all the **CS Core** topics are included;
    b. Maximize the KA Core topics covered;
    c. Eliminate duplicate topics shared with other courses in the curriculum;
    d. Identify the desired **skill level** for each core topic—use the skill levels recommended in the *Table of Core Topics* in Section 3 as a benchmark for comparison;
    e. Create an assessment plan for each course:
        i. Aggregate the **illustrative learning outcomes** of the knowledge units and topics covered by the course;
        ii. Adapt the learning outcomes to the skill levels identified in step 4d.
5. For the knowledge areas not selected for in-depth coverage in step 2 and the knowledge units not selected in step 3:
    a. Design coherent course(s) that cover all the **CS Core** topics in these knowledge areas/units;
    b. Eliminate duplicate topics shared with other courses in the curriculum;
    c. Identify the desired **skill level** for each core topic—use the skill levels recommended in the *Table of Core Topics* in Section 3 as a benchmark for comparison;
    d. Create an assessment plan for each course:
        i. Aggregate the **illustrative learning outcomes** of the knowledge units and topics covered by the course;
        ii. Adapt the learning outcomes to the skill levels identified in step 5d.
6. Complete the curriculum design loop:
    a. Aggregate the course assessment plans from steps 4e and 5d;
    b. Reconcile the aggregation with the competency area(s) identified in step 1.
7. Sequence the courses in the curriculum with prerequisites and corequisites based on the following:
    a. **Mathematics requirements** listed for the courses identified in steps 4 and 5;
    b. Software competency is a prerequisite for most other competency areas.

References

1. ACM/IEEE-CS Joint Curriculum Task Force. “Computing Curricula 2001 Computer Science.” (New York, USA: ACM Press and IEEE Computer Society Press, 2001).
2. ACM/IEEE-CS Interim Review Task Force. “Computer Science Curriculum 2008: An interim revision of CS 2001.” (New York, USA: ACM Press and IEEE Computer Society Press, 2008).
3. ACM/IEEE-CS Joint Task Force on Computing Curricula. “Computing Science Curricula 2013.” (New York, USA: ACM Press and IEEE Computer Society Press, 2013).

4. Sabin, M., Alrumaih, H., Impagliazzo, J., Lunt, B., Zhang, M., Byers, B., Newhouse, W., Paterson, W., Tang, C., van der Veer, G. and Viola, B. Information Technology Curricula 2017: Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology. Association for Computing Machinery, New York, NY, USA, (2017).
5. Clear, A., Parrish, A., Impagliazzo, J., Wang, P., Ciancarini, P., Cuadros-Vargas, E., Frezza, S., Gal-Ezer, J., Pears, A., Takada, S., Topi, H., van der Veer, G., Vichare, A., Waguespack, L. and Zhang, M. Computing Curricula 2020 (CC2020): Paradigms for Future Computing Curricula. Technical Report. Association for Computing Machinery / IEEE Computer Society, New York, NY, USA, (2020).
6. Leidig, P. and Salmela, H. A Competency Model for Undergraduate Programs in Information Systems (IS2020). Technical Report. Association for Computing Machinery, New York, NY, USA, (2021).
7. Danyluk, A. and Leidig, P. Computing Competencies for Undergraduate Data Science Curricula (DS2021). Technical Report. Association for Computing Machinery, New York, NY, USA, (2021).
8. Anderson, L. W. and Krathwohl, D. R., eds. (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. New York: Longman. ISBN 978-0-8013-1903-7.
9. Bamkole, A., Geissler, M., Koumadi, K., Servin, C., Tang, C., and Tucker, C. S., "Bloom's for Computing: Enhancing Bloom's Revised Taxonomy with Verbs for Computing Disciplines". The Association for Computing Machinery. (January 2023). https://ccecc.acm.org/files/publications/Blooms-for-Computing-20230119.pdf, accessed March 2024.