Teacher Licensure option for Computer Science

The state of Illinois has recently placed an emphasis on the teaching and learning of computer science in K-12. The proposed teacher licensure minor will address the Illinois standards for computer science and prepare candidates to teach both regular and AP computer science at the secondary level. The following pages show how the proposed coursework addresses relevant standards for computer science.

Proposed Program

Completion of a teacher licensure minor does not guarantee that the individual will be granted an endorsement to teach in that field. Individuals must meet all requirements (including state tests) as set forth by the Illinois State Board of Education to be granted an endorsement in a second teaching field. Candidates must maintain a

The computer science teacher minor requires completion of the following courses:

CSM 2170 – Computer Science I (4 credits, MAT 1441G co-requisite)
CSM 2670 – Object Oriented Programming (4 credits)
CSM 3670 - Principles of Computer Systems (3 credits)
CSM 3870 - Data Structures (3 credits)
CSM 4880 - Design and Analysis of Algorithms (3 credits)
CSM 3700 – Teaching Computer Science (2 credits)

Recommended: MAT 2345 – Discrete Mathematics (3 credits)

Total = 19 credits

	Courses					
Standards	2170	2670	3670	3870	4880	3470
Demonstrate understanding of the design process (e.g., define the problem; generate ideas; build, test, and improve solutions).	x	x	x	x	x	
Demonstrate understanding of the role of creativity, communication, and collaboration in problem solving.	x	x	x	x	x	
Apply knowledge of connections between elements of mathematics and computer science (e.g., base-two, base-ten, and hexadecimal number systems; logic; sets; functions).			x			
Apply knowledge of how binary sequences represent information (e.g., computer programs, numbers, texts, images).	x	x	x			
Demonstrate knowledge of abstraction to manage problem complexity by using it to decompose problems into subproblems.	x	x	x	x	x	
Use visual representations (e.g., flowcharts, trace tables) to analyze problem states, control structures, and flow of execution, and to identify program outputs.		x	x	x	x	
Apply knowledge of sequence, conditionals, iteration, and recursion as they relate to algorithms.	x	x	x			
Analyze an algorithm to identify and correct errors.	x	x	x	x		
Analyze algorithms for searching, sorting, and finding the minimum, maximum, and/or average.		x			x	
Select or modify a given algorithm to solve a problem.	x	x	x		x	
Evaluate algorithms for efficiency.		x			x	
Apply knowledge of functions/methods and parameters.	x	x	x		x	
Apply the principle of decomposition to a problem by defining new functions/methods and classes.	x	x	x		x	
Apply knowledge of encapsulation and information hiding.	x	x			x	
Apply knowledge of hierarchies and inheritance.		x				
Demonstrate knowledge of polymorphism, composition, and aggregation.		x			x	
Apply knowledge of concepts and principles related to the use of libraries and APIs.	x	x			x	
Demonstrate understanding of Integrated Development Environments (IDEs) and their uses in program development.	x	x			x	
Demonstrate knowledge of computational tools and techniques used to create digital artifacts (e.g., images, animation, video, multimedia, apps).				x		
Demonstrate knowledge of the tools that support program execution (e.g., operating systems, compilers, interpreters).	x	x	x		x	

	Courses					
Standards	2170	2670	3670	3870	4880	3470
Demonstrate knowledge of primitive data types (e.g., Boolean, integer, floating point, string, char).	x	x	x	x	x	
Perform operations on various data types.	x	x		x	x	
Apply properties of lists and arrays to solve problems.	x	x	x	x	x	
Apply knowledge of comparison operators for primitive data types	x	x	x	x	x	
Apply knowledge of Boolean logic	x	x	x	x	x	
Apply principles of conditional control structures (e.g., if statements, switch statements).	x	x	x	x	x	
Apply principles of iterative control structures (e.g., while loops, for loops).	x	x	x	x	x	
Apply principles of recursion.	x	x	x	x	x	
Demonstrate knowledge of the software development process (e.g., design, coding, testing, verification).	x	x	x	x	x	
Demonstrate knowledge of error types (e.g., syntax, runtime, logic).	x	x	x	x	x	
Analyze code segments to identify and correct errors	x	x	x	x	x	
Apply principles of debugging software programs (e.g., adding output statements, hand tracing, using a debugger).	x	x	x	x	x	
Demonstrate understanding of program qualities (e.g., usability, efficiency, portability, scalability).	x	x		x	x	
Demonstrate knowledge of the structure of the Internet and the flow of information through the Internet (e.g., routing, packet switching).						x
Demonstrate knowledge of properties, characteristics, and uses of communication protocols.						x
Demonstrate knowledge of cloud computing and cloud services.						x
Demonstrate knowledge of issues and techniques related to data security and encryption.						x
Demonstrate knowledge of tools (e.g., HTML, formatting and scripting tools) and design techniques used to create Web pages.						x
Demonstrate knowledge of the uses of data collected through the Internet and the tools and techniques for locating, collecting, cleaning, and analyzing the data.						x
						x

	Courses					
Standards	2170	2670	3670	3870	4880	3470
Demonstrate knowledge of issues and practices related to digital citizenship.						x
Demonstrate knowledge of legal and ethical issues related to computing practices (e.g., software privacy, intellectual property rights, scams, data collection and use).						x
Demonstrate understanding of how computing technologies and practices have influenced society (e.g., individual and collective behaviors, enhancing new forms of communication and collaboration, virtual reality, open-source licensing).						x
Demonstrate understanding of how computer science has influenced innovations in art, science, health care, education, and commerce.						x
Demonstrate knowledge of issues related to personal safety and to security and privacy of personal information						x
Demonstrate knowledge of the implications of artificial intelligence, robotics, and microcontrollers						x
Demonstrate knowledge of developing lessons that use effective and engaging practices and methodologies (e.g., inquiry, real-world computing problems, culturally relevant project-based methodologies, encouraging problem solving).	x	x		x	x	
Demonstrate knowledge of effective approaches for promoting collaboration (e.g., group work, peer instruction).	x	x		x	x	
Demonstrate knowledge of approaches for developing effective communication skills	x	x		x	x	
Demonstrate understanding of problematic concepts and constructs in computer science and appropriate strategies to address them	x	x		x	x	
Demonstrate knowledge of environments and activities, including unplugged activities, to foster creativity.	x	x		x	x	
Apply knowledge of instructional strategies, tools, technologies, and practices to support the diverse needs of all learners, paying particular attention to the needs of students from groups underrepresented in computer science (i.e., as defined by race, gender, learning differences).	x	x		x	x	
Demonstrate knowledge of a variety of assessments (e.g., formative, summative, reflective questioning) appropriate for use in computer science.	x	x		x	x	
Demonstrate knowledge of how physical environments can affect learning						x

Department/School Curriculum Committee: 8/27/21
College Curriculum Committee: 9/8/21
Council on Teacher Education: