of startups in any field but especially those in high tech where everything moves so fast. Personally, I don't believe that all of this is a fad that will fade away. But even if you disagree with me, which is fair enough, I would like you to consider the K–12 computing MOOC invasion as a huge opportunity not to be wasted. Don't we ask this exact thing of our students who just want to get through what they see as a distasteful class?

To put it bluntly: the K–12 MOOCs are disruptive, innovative, unconventional, and disrespectful of authority and established power structures. We should be exploring this phenomenon in depth. We should be proactive and open-minded.

Think about it. For many years we have been working hard, really hard, to draw public attention to the need to include computer science in secondary education. We want people to care about coding and to get excited about it. For many years it has felt like our words have been falling on deaf ears. But now? What clearer message could we receive that we have finally begun to be heard than that the technology industry is falling all over itself to get in on the action? We should take the corporate incursions as a compliment and validation of our efforts.

What should we do next with that warm and fuzzy feeling? We have the opportunity to influence a growing movement. The buzz about coding in the startup community provides our opening. Let's look at the situation strategically. We certainly have the smarts, the experience, and the ability to do so. Let's talk about what we can learn from the K–12 computing MOOCs. Challenging circumstances provide some of the best opportunities for learning and growth. Frankly, I am excited by the challenge. Are you? **Ir**

Lisa C. Kaczmarczyk
12710 Torrey Bluff Drive, #176
San Diego, California 92130 USA
lisakacz@gmail.com

DOI: 10.1145/2568195.2568205

COMMUNITY COLLEGE CORNER

Elizabeth K. Hawthorne

CS2013 Exemplar Spotlight on Two-Year Colleges

AFTER THREE INTENSE YEARS of curriculum development work and two draft versions, Strawman and Ironman, the Guidelines for *Computer Science Curricula 2013* were finalized and approved in the fourth quarter of 2013 by both sponsoring professional societies, ACM and IEEE-CS. Hooray! Readers may recall my December 2012 column entitled, "CS2013: A Call for Community College Exemplars" [4]. Throughout the lengthy curriculum development process, the CS2013 Steering Committee called for two categories of exemplars, individual courses and complete degree programs that mapped across the Knowledge Areas (KA) of the CS2013 Body of Knowledge (BoK). The final, 500-page "CS2013 report includes examples of actual fielded courses…to illustrate how topics in the Knowledge Areas may be covered and combined in diverse ways. The report also contains examples of CS curricula from a handful of institutions to show different ways in which a larger collection of courses can be put together to form a complete curriculum [2]."

In this column, I am honored to highlight the selected two-year college exemplars, both courses and associate-degree programs that were mapped to the topics distributed throughout the 18 KAs of the CS2013 BoK. The 18 KAs are: 1) Algorithms and Complexity, 2) Architecture and Organization, 3) Computational Science, 4) Discrete Structures, 5) Graphics and Visual Computing, 6) Human-Computer Interaction, 7) Information Assurance and Security, 8) Information Management, 9) Intelligent Systems, 10) Networking and Communications, 11) Operating Systems, 12) Platform-based Development, 13) Parallel and Distributed Computing, 14) Programming Languages, 15) Software Development Fundamentals, 16) Software Engineering, 17) System Fundamentals, and 18) Social Issues and Professional Practice.

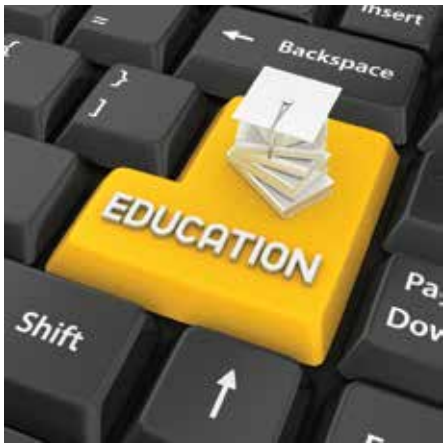

PHOTO: © ISTOCKPHOTO/PAGADESIGN

Two-Year College Course and Curricular Exemplars

All course exemplars are located in Appendix C of the CS2013 final report followed by all curricular exemplars in Appendix D [2]. The following two-year colleges from across the U.S. provided a total of seven exemplars: Anne Arundel Community College in Arnold, Maryland [1], Bluegrass Community and Technical College in

CS2013 Exemplar Spotlight on Two-Year Colleges

TABLE 1: UNION COUNTY COLLEGE COURSE EXEMPLAR: MAT267 DISCRETE MATHEMATICS

KNOWLEDGE AREA	KNOWLEDGE UNIT	TOPICS	HOURS
Discrete Structures	Sets, Relations, Functions	**Sets:** Venn diagrams, union, intersection, complement, Cartesian product, power sets, cardinality of finite sets **Relations:** reflexivity, symmetry, transitivity, equivalence relations, partial orders **Functions:** surjections, injections, bijections, inverses, composition	6
Discrete Structures	Basic Logic	Propositional logic, propositional inference rules, predicate logic, logical connectives, truth tables, normal forms, validity of well-formed formulas	9
Discrete Structures	Proof Techniques	Notions of implication, equivalence, converse, inverse, contrapositive, negation, and contradiction; structure of mathematical proofs; direct proofs; disproving by counterexample; proof by contradiction, induction over natural numbers; structural induction; weak and strong induction; recursive mathematical definitions; well orderings	9
Discrete Structures	Basics of Counting	Counting arguments, pigeonhole principle, permutations and combinations, recurrence relations, basic modular arithmetic	7
Discrete Structures	Graphs and Trees	Trees, tree traversals, undirected graphs, directed graphs, weighted graphs, spanning trees/forests	6
Discrete Structures	Discrete Probability	Finite probability space, events; Axioms of probability; Conditional probability, Bayes' theorem; Independence; Integer random variables; Linearity of Expectation; Variance	5

Lexington, Kentucky [3], Portland Community College in Portland, Oregon [5], and Union County College in Cranford, New Jersey [6].

The spotlight first focuses on the *Discrete Mathematics* course exemplar from Union County College. Discrete Mathematics (MAT267) is a 42 contact-hour, transfer-oriented course that is required for three different associate-degree programs: mathematics, computer science, and computer engineering. Table 1 shows the mapping of this course to the Knowledge Unit topics of the CS2013 Discrete Structures Knowledge Area.

Next in the spotlight is Portland Community College (PCC). Courses at PCC are based on a quarter-credit system (approximately two-thirds of a semester credit), and PCC offers a two-course sequence exemplar in Discrete Structures (CS250 and CS251). Table 2 shows the mapping of these two 30 contact-hour courses to topics in the *Discrete Structures* and

TABLE 2: PORTLAND COMMUNITY COLLEGE COURSE EXEMPLAR: CS250 AND CS251 DISCRETE STRUCTURES I AND II

KNOWLEDGE AREA	KNOWLEDGE UNIT	TOPICS	HOURS CS250 AND CS251
Algorithms and Complexity	Basic Analysis	**CS250:** Differences among best, expected, and worst case behaviors; Big-O, Big Omega, Big-Theta definitions; Complexity classes **CS251:** Empirical measurement and performance; Time and space trade-offs in algorithms; Recurrence relations; Analysis of iterative and recursive algorithms.	4 and 4
Algorithms and Complexity	Basic Automata, Computability and Complexity	**CS251:** Finite state machines, regular expressions, the Halting problem	0 and 4
Discrete Structures	Sets, Relations, Functions	**CS250:** Venn diagrams, union, intersection, complement, Cartesian product, power sets, cardinality, proof techniques **CS251:** Reflexivity, symmetry, transitivity, equivalence relations, partial orders, surjections, injections, bijection,s inverses, composition	4 and 4
Discrete Structures	Basic Logic	CS250: Propositional logic, connectives, truth tables, normal forms, validity, inference, predicate logical, quantification, limitations	10 and 0
Discrete Structures	Proof Techniques	**CS250:** Implications, equivalences, converse, inverse, contrapositive, negation, contradiction, structure, direct proofs, dis-proofs, natural number induction, structural induction, weak/string induction, recursion, well orderings.	10 and 0
Discrete Structures	Basics of Counting	**CS250:** Basic modular arithmetic **CS251:** Counting arguments, cardinality, sum and product rule, IE principle, arithmetic and geometric progressions, pigeonhole principle, permutations, combinations, Pascal's identity, recurrence relations	2 and 10
Discrete Structures	Graphs and Trees	**CS251:** Trees, tree traversals, undirected graphs, directed graphs, weighted graphs, isomorphisms, and spanning trees.	0 and 4
Discrete Structures	Discrete Probability	**CS251:** Finite probability space, events, axioms and measures, conditional probability, Bayes' Theorem, independence, Bernoulli and binomial variables, expectation, variance, conditional independence.	0 and 4

TABLE 3: PORTLAND COMMUNITY COLLEGE COURSE EXEMPLAR: CIS133 JAVA PROGRAMMING I

KNOWLEDGE AREA	KNOWLEDGE UNIT	TOPICS	HOURS
Algorithms and Complexity	Fundamental Data Structures and Algorithms	Simple numerical algorithms, sequential search	3
Programming Languages	Object-Oriented Programming	Classes, objects with state and behavior, encapsulation, visibility, collection classes	8
Programming Languages	Basic Type Systems	A type as a set of values together with a set of operations; Association of types to variables, arguments, results, and fields; Type safety and errors; Generic types in connection with Java collection classes	2
Programming Languages	Language Translation and Execution	Interpretation vs. compilation in connection with the Java language model as contrasted with straight compiled languages (e.g., C++)	1
Software Development Fundamentals	Algorithms and Design	Role of algorithms, problem-solving strategies, fundamental design concepts	2
Software Development Fundamentals	Fundamental Programming Concepts	Basic syntax and semantics of a higher-level language; Variables and primitive data types; Expressions and assignments; Simple I/O including file I/O; Conditional and iterative control structures; Functions and parameter passing	10
Software Development Fundamentals	Fundamental Data Structures	Arrays, records, strings, lists, references, aliasing	12
Software Development Fundamentals	Development Methods	Program comprehension; Program correctness; Simple refactoring; Modern programming environments; Debugging strategies; Documentation and program style	2

Algorithms and Complexity KAs. PCC also provides an introductory programming course exemplar - CIS133 *Java Programming I*. Table 3 shows the mapping of this 40 contact hour lecture/lab course to topics in three different KAs: Algorithms and Complexity, Programming Languages, and Software Development Fundamentals.

The final course exemplar in the spotlight is CSI194 *Ethics in the Information Age* from Anne Arundel Community College. This 41 contact-hour computer science course is cross-listed in the Philosophy department and fulfills a general education core requirement in the state of Maryland. Table 4 shows the mapping of this course to core and elective topics in the Social Issues and Professional Practice KA.

TABLE 4: ANNE ARUNDEL COMMUNITY COLLEGE COURSE EXEMPLAR: CSI194 ETHICS IN THE INFORMATION AGE

KNOWLEDGE AREA	KNOWLEDGE UNIT	TOPICS	HOURS
Social Issues and Professional Practice	Social Context	Social justice, digital divide, distributive justice theories, accessibility issues, social interaction, cultural issues, commerce, free speech and censorship, free speech and hate speech, free speech and pornography, Internet as public or private space, regulatory agencies and laws in physical space, jurisdictional issues with regulating cyberspace	4
Social Issues and Professional Practice	Analytical Tools	Introduction to Ethical Thought: values, morals, normative analysis; Introduction to Cyberethics; Ethical theories: Virtue Ethics, Utilitarianism, Deontology, Just Consequentialism, Social Contract Theory; Evaluate stakeholder positions; Concepts of argumentation and debate	12
Social Issues and Professional Practice	Professional Ethics	Moral responsibility of a professional; Pervasive nature of computing applies to all; Professional Codes of Conduct; Principles of the Joint IEEE-CS/ACM Code of Ethics and Professional Practice; Purpose of a code of ethics; Weaknesses of codes of ethics; Accountability, responsibility and liability	4
Social Issues and Professional Practice	Intellectual Property	Overview and history of intellectual property: trade secrets, patents, trademarks, copyrights; Philosophical views of property: Labor Theory, Utilitarian Theory, Personality Theory; Fair Use; Digital Millennium Copyright Act; Digital Rights management; Alternatives to the property model: GNU project, Open Source Initiative, Creative Commons; Software piracy	6
Social Issues and Professional Practice	Privacy and Civil Liberties	Technology's impact on privacy; Difference between naturally private and normatively private situations; Philosophical foundations of privacy rights; Three types of personal privacy: accessibility, decisional, and informational; different cultural views of privacy; Public and personal information; Information matching technique's impact on privacy; Legal rights to privacy; Solutions for privacy violations	6
Social Issues and Professional Practice	Security Policies, Laws and Computer Crimes	Need to protect computer data, systems, and networks; Ethical issues related to computer security; Social engineering; Identity theft; Computer hacking; Security issues related to anonymity on the Internet; Cyberterrorism and information warfare; Ethical issues related to cyber-related crimes	7

CS2013 Exemplar Spotlight on Two-Year Colleges

In the full curricular exemplar spotlight is Bluegrass Community and Technical College (BCTC). BCTC provides two curricular exemplars by mapping both its associate in science (A.S.) and associate in applied science (A.A.S.) degree programs to the CS2013 BoK. BCTC's A.S. degree in computer science was designed specially to transfer into baccalaureate degree programs. The transferable computer science core is limited to four courses since the degree program also includes all the general education requirements for a baccalaureate degree and is deemed fully "gen ed" certified by the state of Kentucky. The four core courses are: 1) CS 115 Introduction to Computer Programming, 2) CS 215 Introduction to Program Design, Abstraction and Problem Solving, 3) CS 216 Introduction to Software Engineering, and 4) CS 275 Discrete Mathematics.

Typically, A.A.S. degree programs require less general education credits than A.S. degrees programs. In the case of BCTC, fewer general education courses allows for the inclusion of more computing courses in the A.A.S. program. The A.A.S. computer science program at BCTC is comprised of the four core courses of the A.S. degree plus the following additional six courses: 1) CIT 105 Introduction to Computers, 2) CIT 111 Computer Hardware and Software, 3) CIT 150 Internet Technologies, 4) CIT 160 Introduction to Networking Concepts, and 5) CIT 170 Database Design Fundamentals, and 6) CIT 180 Security Fundamentals. These additional courses cover more of the CS2013 core, in particular, the following KAs: architecture and organization (AR), graphics and visual computing (GV), information assurance and security (IAS), information management (IM), networking and communications (NC), operating systems (OS) and social issues and professional practice (SP). Table 5 compares the percentage of core coverage in the CS2013 BoK of both associate-degree programs at BCTC. See Appendix D of the CS2013 Final Report [2] for detailed topic-by-topic mappings for each of these degree programs.

TABLE 5: PERCENTAGE OF CS2013 CORE COVERED BY BCTC PROGRAMS

BCTC DEGREE PROGRAM	COVERAGE OF CORE TIER-1	COVERAGE OF CORE TIER-2
Associate in Science	70%	35%
Associate in Applied Science	77%	52%

I strongly encourage faculty and academic administrators at two-year colleges to compare your institution's computer science curriculum to these exemplars and to the new CS2013 guidelines. It is important to note that there are new KAs to incorporate into your existing degree programs: Information Assurance and Security (IAS), Parallel and Distributed Computing (PD), Software Development Fundamentals (SDF) and Systems Fundamentals (SF). The discipline of computer science is constantly changing, and the CS2013 Curricular Guidelines will provide educators with a navigational beacon into the next decade. lr

References

[1] Anne Arundel Community College. http://www.aacc.edu/. Accessed 2013 November 15.

[2] Association for Computing Machinery and IEEE Computer Society. (October 2013). "Computer Science Curricula 2013 Final Report 0.9 (pre-release version)." http://cs2013.org/final-draft/CS2013-Final-v0.9-prerelease.pdf. Accessed 2013 November 15.

[3] Bluegrass Community and Technical College. http://www.bluegrass.kctcs.edu/. Accessed 2013 November 15.

[4] Hawthorne, E. K. (December 2012). "CS2013: A Call for Community College Exemplars." *ACM Inroads*, 3(4), 24-25. DOI: 10.1145/2381083.2381094.

[5] Portland Community College. http://www.pcc.edu/. Accessed 2013 November 15.

[6] Union County College. http://www.ucc.edu/. Accessed 2013 November 15.

Elizabeth K. Hawthorne
Computer Science Department
Union County College
Cranford, New Jersey 07016 USA
Hawthorne@ucc.edu

DOI: 10.1145/2568195.2568206