1-11-2015

Computer Science & Engineering Academic Program Review, 2014

College of Engineering & Computer Science, Wright State University

College:

Engineering and Computer Science

Department:

Computer Science & Engineering

Academic Programs Reviewed:

Bachelors of Arts in Computer Science (BACS)

Bachelors of Science in Computer Science (BSCS)

Bachelors of Science in Computer Engineering (BSCE)

Master of Science in Computer Science (MSCS)

Master of Science in Computer Engineering (MSCE)

Master of Science in Cyber Security (MSCyS)

Program Review Committee:

Travis Doom, Associate Professor and Associate Chair, Computer Science & Engineering

Arthur Goshtasby, Professor and Graduate Program Director, Computer Science & Engineering

Karen Meyer, Senior Lecturer and Undergraduate Program Director, Computer Science & Engineering

T.K. Prasad, Professor and CSE Graduate Studies Committee Chair

Vance Saunders, Instructor and Cyber Security Program Director

Submitted: January, 2015

Mateen Rizki, Chair _____

Nathan Klingbiel, Dean _____

Bachelors of Arts in Computer Science – BACS

Enrollment and Graduate History

	Fall 09	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Enrollment	29	44	36	46	45
Graduates	9	11	9	12	13

Program description

The Bachelor of Arts in Computer Science program prepares students for careers in computer systems analysis and design, software development, system administration and web site development. The Bachelor of Arts program allows for maximum flexibility of course study with fewer requirements in high level math and science. Our Graduates can pursue a wide array of careers and develop a broad understanding of multiple disciplines and the application of technology and computer systems in these fields.

[PEO EXPERT] Graduates of the Computer Science BA program are employable as computing professionals and will be recognized by their employers as well-prepared for their career in computing. [PEO AGILE] Graduates understand that education is a lifelong process and are well prepared for continuing studies. [PEO ENGAGED] Graduates demonstrate appreciation for the professional, social, ethical and leadership roles of computing professionals. [PEO APPLIED] Graduates can apply computing and software development principles to a diverse range of domains, such as analytics, data science, informatics, management, etc.

Alignment with university mission, strategic plan

The Computer Science BA program educational objectives (PEOs) are in alignment with the university's core mission. The BA CS program builds a solid foundation for student success by preparing students with the critical thinking, mathematical, communication, and technical skills necessary for meaningfully engaging both as informed citizens and in the workplace [PEOs EXPERT, APPLIED]. The BA CS program prepares students to enter into high-demand technical careers that help drive economic growth in the region/state [PEOs EXPERT, APPLIED]. Most importantly, the BA CS program prepares students for the process of lifelong education so that they can more easily continue to develop professionally, intellectually, and personally after graduation [PEOs AGILE, ENGAGED].

Program distinctiveness

The Computer Science BA program shares most of its outstanding features with the department's externally accredited Computer Science BS and Computer Engineering BS programs. The distinguishing feature of the CS BA program is a focus on the application of computing to a diverse range of domains. The CS BA program requires less mandatory course work towards developing mathematically rigorous scientific-method based skills. Instead, the CS BA program provides opportunity for students to develop knowledge in other domains in which computing is applied. These potential domains of interest include not only the sciences, but also business, the arts, health care, or any other area of university study.

The BACS program has recently incorporated a new "fast track" advising program designed to provide a BACS degree through 18 months of intensive study in the discipline of computer science to potential students that already possess a baccalaureate degree in another discipline. In addition, the BACS program has a recently implemented option to provide professional K-12 Teacher Licensure. All BACS majors (including pre- and intending majors) are advised by professional advisors within the college of Engineering and Computer Science.

Recognitions of quality of the program

This program is supported by the same faculty, core course sequence, and assessment infrastructure as the department's externally accredited (ABET) Bachelors of Science programs in Computer Science and in Computer Engineering. As such, there is no doubt that the graduates of this program receive a high quality experience similar in outcome to our BS programs.

Program learning outcomes

Students who complete the Bachelor of Arts in Computer Science will have:

1. an ability to apply knowledge of mathematics.
2. an ability to apply knowledge of science.
3. an ability to apply knowledge of theory of computation.
4. an ability to design and conduct experiments.
5. an ability to analyze and interpret data and report the results of the interpretation.
6. an ability to identify, formulate, and solve computer oriented problems as appropriate to the discipline of computer science.
7. an ability to design software to meet desired needs.
8. an ability use the techniques, skills, and modern tools necessary for professional practice.
9. an ability to communicate effectively in written, graphical and oral forms.
10. an understanding of professional and ethical responsibility.
11. a knowledge of contemporary issues: social and ethical, as well as technical issues in local, regional, national and international context.
12. the broad education necessary to understand the impact of science and technology in a global and societal context: relevant to being a good citizen at the local, national, and international levels.

Description of description of learning outcomes assessment program

The Computer Science Department Faculty have produced a mapping of Knowledge Topics prerequisite or developed in each of the core/mandatory courses in our program. Achievements in these Knowledge Topics are assessed in subsequent courses. A direct assessment of Knowledge Topics is made for every student enrolled in every core course in the program every term. These direct assessments take place as on-line prerequisite knowledge assessments at the beginning of courses that utilize and build on that knowledge topic. SLOs are mapped to specific knowledge topics developed in specific core courses. These direct assessments form the examination basis for program SLOs. Additionally, indirect assessments are obtained from two formal groups, the department's external advisory board and the department's student advisory board.

Summary of assessment findings for past five years

The semester-based version of this program began Fall 2012. Given the short period of data collection for semester-based program of study, very few significant actions have been taken on the collected assessment data. Some potential concerns have been noted in our annual assessment report and flagged for long-term observation. Please find the 2014 Departmental Assessment findings attached.

Major curricular changes since last review (or past five years)

The 2012-2013 academic year saw a complete redesign of all three undergraduate computer science & engineering programs due to the university-wide transition to semester-based terms. As no assessment data exists for the new semester-based courses or programs of study, assessment efforts during the 2012-2013 cycle has been largely focused on the development of direct assessment instruments and collection of data for the newly offered programs/courses.

Three primary initiatives have been taken to improve student learning during this cycle:

- *Delivery of inverted-lecture core sequence (SCALE-UP Classrooms - 152 RC & 355 RC)*
- *Development of program educational objectives with program constituents*
- *Preliminary development of infrastructure for continuous assessment of relevant retained knowledge*

Graduate placement data, employer satisfaction

The BA program is relatively new and does not yet have graduates with a “five-year out” history of employment. Starting in 2014, graduating students are polled on expected placement. This effort will help bridge the gap on assessing graduate placement until a significant and experienced cohort of students have been employed in the field for a duration appropriate to gauge employer satisfaction. Anecdotal evidence suggests that graduates of the Computer Science BA program are receiving many of the same job offers as graduates of the Computer Science BS program and with similarly positive employer satisfaction.

If program has professional accreditation, attach most recent review findings and recommendations

NA

Bachelors of Science in Computer Science – BSCS

Enrollment and Graduate History

	Fall 09	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Enrollment	305	298	354	349	403
Graduates	29	41	47	54	50

Program description

The Department of Computer Science and Engineering has been part of the College of Engineering and Computer Science at Wright State University since the college was founded in 1986. Prior to that time the department's programs were organized in the College of Science and Mathematics. Our Bachelor of Science in Computer Science (BSCS) degree program has a long history. It was established in 1968 – a time when computer science was just becoming recognized as a major discipline in universities across the country. The program has been accredited by ABET continuously since 1987. The last general review of the BSCS program occurred in AY 2010-2011.

The Bachelor of Science in Computer Science degree offers a curriculum in the study of the software aspects of computer systems including the study of algorithms and data structures, programming languages, software methodology and tools, data management and analysis. [PEO EXPERT] Graduates of the Computer Science program are employable as computing professionals and will be recognized by their employers as well-prepared for their career in computing. [PEO AGILE] Graduates understand that education is a lifelong process and are well prepared for continuing studies, including graduate studies. [PEO ENGAGED] Graduates demonstrate appreciation for the professional, social, ethical and leadership roles of computing professionals. [PEO FOCUSED] Graduates have a set of software theory and development skills that emphasizes software construction, team-based project management, and experience with contemporary software development tools/paradigms.

Alignment with university mission, strategic plan

The BSCS program educational objectives are consistent with the mission of Wright State University. The University's goal of "achieving learning outcomes through innovative, high quality programs for all students" is well supported by PEO EXPERT. Further, Wright State University's commitment to "conducting scholarly research and creative endeavors" enables the faculty to remain on the leading edge of the practice, and to continue to provide students with experiences developed toward meeting PEOs EXPERT and FOCUSED. WSU's commitment to "engaging in significant community service" provides a clear example of social responsibility and ethical practice to our students, in support of PEO ENGAGED. Finally, PEOs AGILE and ENGAGED are essential elements in achieving the overall aim of the institution, to "transform the lives of our students and the communities we serve".

Program distinctiveness

1. The program is housed in the same department as the Computer Engineering program making it easier for students to pursue both majors and/or enroll in computer science as well as computer engineering elective courses.

2. Wright State University is located near Wright Patterson Air Force Base providing students with unique opportunities to work in the Air Force Research Lab and other related employment opportunities.

3. The Boffin Factory is a student centered research, activity and study area where students can pursue specific interests, connect with other students and attend lectures hosted by faculty and staff.

4. The program is small enough that students can be involved in working in help rooms, labs and recitations and have direct contact with their Professors, but large enough that they have access to resources provided by the defense and private industry.

The BSCS program has recently incorporated a new "fast track" advising program designed to provide a BSCS degree through 18 months of intensive study in the discipline of computer science to potential students that already possess a baccalaureate of degree in another science. In addition, the BSCS program has a recently implemented a "4+1" track that allows high achieving students to being progress towards a graduate degree in computer science while completing senior-level electives in their baccalaureate program of study. All BSCS majors (including pre- and intending majors) are advised by professional advisors within the college of Engineering and Computer Science.

Recognitions of quality of the program

The program has been accredited by ABET/CAC continuously since 1987.

Program learning outcomes

Students who complete the BS in computer science will have:

1. an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices.
2. an ability to design and conduct experiments, coupled with an ability to analyze and interpret data, and report the results of the interpretation.
3. an ability to apply design and development principles to design, implement, and evaluate software systems (computer-based systems, processes, components, or programs) of varying complexity to meet desired needs.
4. an ability to function effectively on teams to accomplish a common goal.
5. an ability to identify, formulate, and solve computer oriented problems as appropriate to the discipline of computer science.
6. an understanding of professional, ethical, legal, security and social issues and responsibilities.
7. an ability to communicate effectively in written (prose as well as mathematical, scientific,and engineering notations in technical reports), graphical (diagrams, charts, visualizations, animations), and oral (discussions with colleagues, group meetings, and formal presentations) forms.

8. the broad education necessary to understand the impact of science and technology in a contemporary global and societal context: relevant to being a productive and contributing citizen at the local, national, and international levels.
9. a recognition of the need for, and an ability to engage in life-long learning of computer science and related topics.
10. an ability to use the techniques, skills, and modern tools necessary for professional computing practice such as software development environments, modern programming languages, and computer hardware components.

Description of description of learning outcomes assessment program

The Computer Science Department Faculty have produced a mapping of Knowledge Topics prerequisite or developed in each of the core/mandatory courses in our program. Achievement in these Knowledge Topics are assessed in subsequent courses. A direct assessment of Knowledge Topics is made for every student enrolled in every core course in the program every term. These direct assessments take place as on-line prerequisite knowledge assessments at the beginning of courses that utilize and build on that knowledge topic. SLOs are mapped to specific knowledge topics developed in specific core courses. These direct assessments form the examination basis for program SLOs. Additionally, indirect assessments are obtained from two formal groups, the department's external advisory board and the department's student advisory board.

Summary of assessment findings for past five years

The semester-based version of this program began Fall 2012. Given the short period of data collection for semester-based program of study, very few significant actions have been taken on the collected assessment data. Some potential concerns have been noted in our annual assessment report and flagged for long-term observation. Please find the 2014 Departmental Assessment findings attached.

Major curricular changes since last review (or past five years)

The 2012-2013 academic year saw a complete redesign of all three undergraduate computer science & engineering programs due to the university-wide transition to semester-based terms. As no assessment data exists for the new semester-based courses or programs of study, assessment efforts during the 2012-2013 cycle has been largely focused on the development of direct assessment instruments and collection of data for the newly offered programs/courses.

Three primary initiatives have been taken to improve student learning during this cycle:

- *Delivery of inverted-lecture core sequence (SCALE-UP Classrooms - 152 RC & 355 RC)*
- *Development of program educational objectives with program constituents*
- *Preliminary development of infrastructure for continuous assessment of relevant retained knowledge*
- *Incorporation of mandatory Team Projects senior capstone experience*

Graduate placement data, employer satisfaction

There have been no significant reports of CS students having difficulty finding immediately placement locally or national. The CS skillset is in high demand and market forces seem to indicate significant

continued demand for computer science graduates. Employer satisfaction (as communicated by the departmental external advisory board) remains high.

If program has professional accreditation, attach most recent review findings and recommendations

Attached.

Bachelors of Science in Computer Engineering – BSCE

Enrollment and Graduate History

	Fall 09	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Enrollment	227	220	235	242	248
Graduates	17	14	17	31	30

Program description

The Department of Computer Science and Engineering has been part of the College of Engineering and Computer Science at Wright State University since the college was founded in 1986. Prior to that time the department's programs were organized in the College of Science and Mathematics. Our Bachelor of Science in Computer Science degree program was established in 1968 - a time when computer science was just becoming recognized as a major discipline in universities across the country. Our Bachelor of Science in Computer Engineering (BSCEG) was established in 1981. The program has been accredited by ABET continuously since 1987. The last general review of the BSCEG program occurred in AY 2010-2011.

The Bachelor of Science in Computer Engineering degree offers a curriculum in the study of computer systems including the design, construction, and overall operations with a major focus on hardware. It includes the study of digital circuits, embedded systems programming languages and operating systems. The program provides a solid mathematics, basic science, and engineering science base that is common to all quality engineering programs. [PEO EXPERT] Graduates of the Computer Engineering program are employable as computing professionals and will be recognized by their employers as well-prepared for their career in computing. [PEO AGILE] Graduates understand that education is a lifelong process and are well prepared for continuing studies, including graduate studies. [PEO ENGAGED] Graduates demonstrate appreciation for the professional, social, ethical and leadership roles of computing professionals. [PEO BROAD] Graduates have a knowledge of computing principles that includes all levels of modern computational architectural/infrastructure, operating systems and component-based hardware/embedded/software systems.

Alignment with university mission, strategic plan

The BSCEG program educational objectives are consistent with the mission of Wright State University. The University's goal of "achieving learning outcomes through innovative, high quality programs for all students" is well supported by PEO EXPERT. The program aims to maintain a modern mix of hardware and software experiences that keep up with the rapidly-changing environment of contemporary computing. Further, Wright State University's commitment to "conducting scholarly research and creative endeavors" enables the faculty to remain on the leading edge of the practice, and to continue to provide students with the modern laboratory, project, and design experiences mentioned in PEOs EXPERT and BROAD. WSU's commitment to "engaging in significant community service" provides a clear example of social responsibility and ethical practice to our students, in support of PEO ENGAGED. Finally, PEOs AGILE and ENGAGED are essential elements in achieving the overall aim of the institution, to "transform the lives of our students and the communities we serve".

Program distinctiveness

1. The program is housed in the same department as the Computer Engineering program making it easier for students to pursue both majors and/or enroll in computer science as well as computer engineering elective courses.

2. Wright State University is located near Wright Patterson Air Force Base providing students with unique opportunities to work in the Air Force Research Lab and other related employment opportunities.

3. The Boffin Factory is a student centered research, activity and study area where students can pursue specific interests, connect with other students and attend lectures hosted by faculty and staff.

4. The program is small enough that students can be involved in working in help rooms, labs and recitations and have direct contact with their Professors, but large enough that they have access to resources provided by the defense and private industry.

The BSCEG program has a recently implemented a "4+1" track that allows high achieving students to being progress towards a graduate degree in computer engineering while completing senior-level electives in their baccalaureate program of study. All BSCEG majors (including pre- and intending majors) are advised by professional advisors within the college of Engineering and Computer Science.

Recognitions of quality of the program

The program has been accredited by ABET continuously since 1987.

Program learning outcomes

Students who complete the BS in computer engineering will have:

1. an ability to apply knowledge of mathematics, science, and engineering.
2. an ability to design and conduct experiments as needed to evaluate artifacts and processes not suitable to other analysis, coupled with an ability to analyze and interpret data possibly using statistical, logical, inductive, graphical, analogical, etc. reasoning and report the results of the interpretation.
3. an ability to design a system, component, or process to meet desired needs.
4. an ability to function on multidisciplinary teams such as in group projects.
5. an ability to identify, formulate, and solve engineering and science problems as appropriate to the discipline of computer engineering.
6. an understanding of professional and ethical responsibility.
7. an ability to communicate effectively in written (prose as well as mathematical, scientific, and engineering notations in technical reports), graphical (diagrams, charts, visualizations, animations), and oral (discussions with colleagues, group meetings, and formal presentations) forms.
8. the broad education necessary to understand the impact of engineering and scientific solutions in a contemporary global , economic, environmental, and societal context: relevant to being a productive and contributing citizen at the local, national, and international levels.

9. a recognition of the need for, and an ability to engage in life-long learning of computer engineering and related topics.
10. an ability to use the techniques, skills, and modern tools necessary for professional engineering practice such as CAD tools and physical instruments, modern programming languages, and computer hardware components.

Description of description of learning outcomes assessment program

The Computer Science Department Faculty have produced a mapping of Knowledge Topics prerequisite or developed in each of the core/mandatory courses in our program. Achievement in these Knowledge Topics are assessed in subsequent courses. A direct assessment of Knowledge Topics is made for every student enrolled in every core course in the program every term. These direct assessments take place as on-line prerequisite knowledge assessments at the beginning of courses that utilize and build on that knowledge topic. SLOs are mapped to specific knowledge topics developed in specific core courses. These direct assessments form the examination basis for program SLOs. Additionally, indirect assessments are obtained from two formal groups, the department's external advisory board and the department's student advisory board.

Summary of assessment findings for past five years

The semester-based version of this program began Fall 2012. Given the short period of data collection for semester-based program of study, very few significant actions have been taken on the collected assessment data. Some potential concerns have been noted in our annual assessment report and flagged for long-term observation. Please find the 2014 Departmental Assessment findings attached.

Major curricular changes since last review (or past five years)

The 2012-2013 academic year saw a complete redesign of all three undergraduate computer science & engineering programs due to the university-wide transition to semester-based terms. As no assessment data exists for the new semester-based courses or programs of study, assessment efforts during the 2012-2013 cycle has been largely focused on the development of direct assessment instruments and collection of data for the newly offered programs/courses.

Three primary initiatives have been taken to improve student learning during this cycle:

- *Delivery of inverted-lecture core sequence (SCALE-UP Classrooms - 152 RC & 355 RC)*
- *Development of program educational objectives with program constituents*
- *Preliminary development of infrastructure for continuous assessment of relevant retained knowledge*

Graduate placement data, employer satisfaction

There have been no significant reports of CS students having difficulty finding immediately placement locally or national. The CS skillset is in high demand and market forces seem to indicate significant continued demand for computer science graduates. Employer satisfaction (as communicated by the departmental external advisory board) remains high.

If program has professional accreditation, attach most recent review findings and recommendations

Attached.

Master of Science in Computer Science

Enrollment and Graduate History

	Fall 2009	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Enrollment	49	47	42	36	106
Graduates	24	14	27	10	15

Program Description

The program offers a wide range of courses in computer science and the opportunity to develop research skills in computer science areas. The program's strengths include diverse faculty expertise, various computer science laboratories, and a balance of theory and practice. Degree requirements focus on the areas of software systems design and analysis, and computer science theory. Courses for the program are offered mostly in the late afternoon and evening hours and some with added online sections to serve the educational needs of practicing computer professionals.

Alignment with the University Mission and Strategic Plan

The program provides students with the solid educational foundation in advanced topics in computer science necessary to support the critical needs of employers in our region and state. The scholarly research of the faculty provides opportunities for students to explore emerging topics at the frontier of computer science to support future economic development and entrepreneurial enterprises.

Program Distinctiveness

- Courses in emerging disciplines such as soft computing, data mining, machine learning, network science, information retrieval, and semantic web.
- Thesis options in wide computer science areas.
- Evening and online courses.
- The option to take courses in computer engineering and cyber security as a part of the graduation requirement.

Recognitions of Quality of the Program

- The program has been steadily growing since its inception in Sept. 1975.
- Supported by government and private sources for research in computer science areas of national and local interest.
- Faculty publications appearing in top science and engineering journals and conferences.
- Program graduates being placed at various positions locally and nationally, some very competitive.
- Adds Big and Smart Data Sciences educational certificate to provide a detailed technical overview of Big Data analysis issues for working professionals.

Program Learning Outcomes

Graduates are able to demonstrate:

- The ability to integrate and apply graduate computer science knowledge to solve complex computer science problems.

- The ability to understand and integrate new knowledge within the field of computer science into their daily professional activities.
- The ability to recognize the need for, and engage in, life-long learning.

Description of Learning Outcomes Assessment Program

To ensure that the Computer Science M.S. program meets the needs of our graduate students and their employers, an annual assessment of the program is planned to evaluate the program's strengths and weaknesses, specifically in regard to the number of annual graduations, time to graduation, satisfaction of graduates with the program, and preparedness of the graduates in tackling technical challenges at work. An indicator of the program's health is the ratio of students to faculty in the program. To measure the preparedness of the graduates in research and tackling difficult problems, the ratio of graduates seeking a doctoral degree is used.

Summary of Assessment Findings for the Past Five Years

Due to the increased need for software development in various engineering, scientific, and commercial applications, job opportunities in computer science have been robust. Consequently, to meet the needs of an increased number of applicants in computer science, during recent years the department has hired a number of new faculty in emerging fields. The CS program at Wright State is currently considered one of the strongest in the State of Ohio.

During recent years, the department has made considerable investments renovating instructional labs, developing multimedia tools for teaching, and enhancing delivery techniques. The program has competent faculty with diverse expertise, who have developed a strong curriculum in the program as evidenced by the exit surveys of the graduates.

Major Curricular Changes During the Past Five Years

- The quarter system program was converted to the current semester system program.
- New courses were introduced to the program as new faculty joined the program.
- Existing course contents were updated as needed to reflect recent advances in computer science.

Graduate Placement Data and Employer Satisfaction

Among the students that graduated between 2010 and 2013, 45% have stated that they already have a position either at AFRL or at a company in the Dayton area and intend to continue working there after graduation. About 40% of the graduates seek positions and some find employment before or shortly after graduation. Many of the students who are not already employed represent international students who will be doing practical training after graduation and will be looking for a position at a later date. The graduates in this category may take jobs outside of Dayton and leave the area. The remaining 15% of the graduates enter the Ph.D. program at Wright State.

Employers of the graduates of the program who are members of the department's advisory board have expressed satisfaction with the graduates.

If Program has Professional Accreditation, Attach Most Recent Findings and Recommendations

N/A

Program 2: Master of Science in Computer Engineering

Enrollment and Graduate History

	Fall 2009	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Enrollment	25	27	13	14	24
Graduates	22	14	10	11	12

Program Description

The program offers diverse courses in computer engineering and the opportunity to develop research skills in computer engineering areas. The program's strengths include a wide range of faculty expertise, many computer engineering laboratories, and a balance of theory and practice. Degree requirements include hardware and software systems design and analysis. Courses for the program are offered mostly in the late afternoon and evening hours and some with added online sections to serve the educational needs of practicing computer professionals.

Alignment with the University Mission and Strategic Plan

The program provides students with the solid educational foundation in advanced topics in computer engineering necessary to support the critical needs of employers locally and within the state. The scholarly research of faculty provides opportunities for students to explore emerging topics at the frontier of computer engineering to support future economic development and entrepreneurial enterprises.

Program Distinctiveness

- Courses in emerging disciplines such as embedded systems, distributed computing, mobile computing, cloud computing, and computer vision.
- Thesis options in wide computer engineering areas.
- Evening and online courses.
- The option to take courses in computer science and cyber security as a part of the graduation requirement.

Recognitions of Quality of the Program

- The program has been steadily growing since its inception in Sept. 1984.
- Supported by government and private sources for research in computer engineering areas of national and local need.
- Faculty publications appearing in top science and engineering journal and conferences.
- Program graduates being placed at various positions locally and nationally, some very competitive.

Program Learning Outcomes

Graduates are able to demonstrate:

- The ability to integrate and apply graduate computer engineering knowledge to solve complex computer engineering problems.
- The ability to understand and integrate new knowledge within the field of computer engineering into their professional activities.
- The ability to recognize the need for, and engage in, life-long learning.

Description of Learning Outcomes Assessment Program

To ensure that the Computer Engineering M.S. program meets the needs of our graduate students and their employers, an annual assessment of the program is planned to evaluate the program's strengths and weaknesses, specifically to determine the number of annual graduations, time to graduation, satisfaction of graduates with the program, and preparedness of the graduates in tackling technical challenges at work. An indicator of the program's health is the ratio of students to faculty in the program. To measure the preparedness of the graduates in research and tackling difficult problems, the ratio of graduates seeking a doctoral degree is used.

Summary of Assessment Findings for the Past Five Years

During recent years, the department has made considerable investments renovating instructional labs, developing multimedia tools for teaching, and enhancing delivery techniques. The program has competent faculty with diverse expertise, offering a strong curriculum as evidenced by the exit surveys of the graduates.

Major Curricular Changes During the Past Five Years

- The quarter system program was converted to the current semester system.
- New courses were introduced to the program as new faculty joined the program.
- Existing course contents were updated as needed to reflect recent advances in computer engineering.

Graduate Placement Data and Employer Satisfaction

Among the students that graduated between 2010 and 2013, 50% stated that they already have a position either at AFRL or at a company in the Dayton area and intend to continue working there after graduation. 42% of graduates seek employment and the majority find employment before or shortly after graduation. International students usually start practical training after graduation for about a year and look for a job after that. The graduates in this category may leave the Dayton area if the job they find is outside Dayton. The remaining 8% of the graduates enter the Ph.D. program or another degree program at Wright State.

Employers of the graduates of the program who are members of the department's advisory board have expressed satisfaction with the graduates.

If Program has Professional Accreditation, Attach Most Recent Findings and Recommendations

N/A

Program 3: Master of Science in Cyber Security

Enrollment and Graduate History:

	Fall 2010	Fall 2011	Fall 2012	Fall 2013	Fall 2014
Enrollment	4	8	4	8	17
Graduates	0	0	0	0	1

Program Description

The program offers a wide range of courses in cyber security and the opportunity to develop research skills in the field. The program strengths include a unique blend of faculty expertise, well-equipped laboratory facilities, and a balance of theory and practice. The degree is focused on developing the knowledge and skills applicable to protecting complex systems operating in cyberspace. Courses are offered online and in residence.

Alignment with the University Mission and Strategic Plan

The program provides students with the solid educational foundation in advanced topics in cyber security necessary to support the critical needs of employers in our region and the state. The scholarly research of the faculty provides opportunities for students to explore emerging topics at the frontier of cyber security to support future economic development and entrepreneurial enterprises.

Program Distinctiveness

- Adds Cyber Security Analytics educational certificate to provide a detailed technical overview of cyberspace for working professionals in other disciplines and fields. These students do not need an M.S. in Cyber Security; however, they do need a technical understanding of the impacts cyber security may have on their specific professional discipline.
- Significantly expands the traditional internet/web definition of cyberspace providing students with a more accurate frame of reference to address cyber security issues. This expanded definition encompasses complex systems for air, land, sea, space, critical infrastructure and The Internet of Things (IoT).
- Establishes a Cooperative Research and Development Agreement (CRADA) with the Air Force Research Laboratory (AFRL) at Wright Patterson Air Force Base (WPAFB) to support the expanded definition of cyberspace involving aircraft and avionics systems.
- Establishes collaboration with the Air Force Institute of Technology (AFIT) to focus on analyzing cyber attacks and defenses across a wide range of complex cyber physical systems.
- Provides extensive hands-on laboratory exercises conducted in our Virtual Cyber Security Lab (VCSL) using current attack and defend tools and techniques.

Recognitions of Quality of the Program

- The program continues to grow since its inception in September 2012.
- It is of great interest to AFRL researchers and employees of local government contractors.

Program learning outcomes:

- The ability to integrate and apply graduate cybersecurity knowledge to solve complex cybersecurity issues and challenges.
- The ability to understand and integrate new knowledge within the field of cybersecurity into their professional activities.

- The ability to recognize the need for, and engage in, life-long learning.
- Obtain a deeper understanding of the breadth and depth of cyberspace and the inefficiencies and shortcomings of our existing evaluation systems to deal with cybersecurity threats.
- Understand the unique characteristics of cyberspace and how these unique characteristics affect/influence cybersecurity threats.
- Identify social, political, and economic factors/impacts of cyber threats and be able to identify and discuss ethical issues related to cybersecurity and privacy.
- Recognize the basic concepts of cyber security defense and be able to use software tools for malware identification and elimination, data encryption and transmission, and key-based authentication.

Description of Learning Outcomes Assessment Program

No assessment is yet available due to the short life of this program in the department.

Summary of Assessment Findings for the Past Five Years

N/A

Major Curricular Changes During the Past Five Years

Course CEG 6424 “Security Attacks and Defenses” was newly added as a required course – changing the total required courses from 3 to 4.

Graduate Placement Data and Employer Satisfaction

N/A

If Program has Professional Accreditation, Attach Most Recent Findings and Recommendations

N/A

Departmental Summary: Computer Science & Engineering

Faculty demographics

	2008	2009	2010	2011	2012
Full	7	8	8	8	9
Associate	6	6	6	6	9
Assistant	5	5	5	5	2
Inst/Lect	5	5	5	6	6
Total	23	24	24	25	26

Staffing Summary

	2008	2009	2010	2011	2012
Unclassified	2	2	5	4	4
Classified	3	3	3	3	3
Total	5	5	8	7	7

Student/Faculty Ratio

	2008	2009	2010	2011	2012
Student FTE/Fac FTE	14	14	15	13	13

Average Class Size

	2010	2011	2012
Lecture	34	20	27
Lab only	25	14	20
Lecture/Lab	42	23	27

Student Data for All Programs and for Graduate Programs in Unit

	Fall 09	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Enrollment	653	649	701	719	855
Graduate	99	104	119	124	131

Total Courses Taught vs. Credit Hours Generated for Unit

	Fall 09	Fall 2010	Fall 2011	Fall 2012	Fall 2013
Undergraduate	218 / 13,465	241 / 13,871	254 / 12,990	145 / 11,492	164 / 12,786
Graduate	195 / 2,293	235 / 2,339	240 / 2,198	226 / 2,169	251 / 3,630
Total	413 / 15,758	476 / 16,210	494 / 15,188	371 / 13,661	415 / 16,416

Course Completions

	2008	2009	2010	2011	2012
Undergraduate	79%	79%	78%	79%	78%
Master's	92%	91%	96%	95%	93%

Expense per Student and Revenue to Expense Ratio

	2008	2009	2010	2011	2012
Expense per Student	$6307	$7126	$7307	$8499	$9928
Rev/Expense	2.77	2.30	2.23	1.86	1.79

Research and External Funding

	2008	2009	2010	2011	2012
External Funding	$1,452,949	$1,337,741	$2,099,268	$1,858,682	$1,207,236

Future employment projections for discipline (to be provided to unit)

Employment prospects for graduates in computer science, computer engineering, and cyber security are very good. Demand for graduates in computer science and computer engineering has been strong and is expected to remain strong for years to come due to expected continued demand in data analysis and digital communications. The need for the graduates in cyber security is especially acute as such expertise is in great demand in government and private agencies and this demand is expected to remain strong for many years to come. Graduates in computer science, computer engineering, and cyber security are expected to find jobs within their field of specialty without any difficulty.

During the recent recession, national employment for computer scientists did not rise above 6% and returned to pre-recession levels well before average unemployment indicators. Employment projects for common career paths for computer scientists are included below. National data on common career paths is collected from the Occupational Outlook Handbook, Bureau of Labor Statistics (http://www.bls.gov, Oct 2014). Quick Facts are based on data collected by BLS in 2012.

- *Computer and information system managers*: $120,850. Entry level education: Bachelor's degree; Number of jobs: 332,700; Job Outlook 2012-2022: +15% (Faster than average)
- *Computer programmers*; Median Pay: $74,280. Entry level education: Bachelor's degree; Number of jobs: 343,700; Job Outlook 2012-2022: +8% (As fast as average)
- *Computer systems analysts*; Median Pay: $79,680. Entry level education: Bachelor's degree; Number of jobs: 520,600; Job Outlook 2012-2022: +25% (Much faster than average)
- *Database Administrators*; Median Pay: $77,080. Entry level education: Bachelor's degree; Number of jobs: 118,700; Job Outlook 2012-2022: +15% (Faster than average)
- *Information security analysts*; Median Pay: $86,170. Entry level education: Bachelor's degree; Number of jobs: 75,100; Job Outlook 2012-2022: +37% (Much faster than average)
- *Software developer*; Median Pay: $94,350. Entry level education: Bachelor's degree; Number of jobs: 1,018,000; Job Outlook 2012-2022: +22% (Much faster than average)

- *Web developers*; Median Pay: $62,500. Entry level education: Associate's degree; Number of jobs: 141,400; Job Outlook 2012-2022: +20% (Faster than average)

Description of how unit programs and curricula are "mission critical" to the core Wright State educational experience

It is the mission of Wright State University to transform the lives of students and the community it serves by: 1) building a solid foundation for student success at all levels through high-quality, innovative programs; 2) conducting scholarly research and creative endeavors that impact the quality of life; 3) engaging in meaningful community service; 4) driving the economic revitalization of our region and our state; and 5) empowering all of our students, faculty, staff, and alumni to develop professionally, intellectually, and personally.

The computer science, computer engineering, and cyber security curriculums help the university meet its mission by offering very competitive programs in disciplines that are in great demand and by graduating competent individuals who can then help the organizations they join to sharpen their technological edge. This will, in turn, revitalize the local and state economy through the delivery of numerous capabilities that are in great demand today.

Technology has changed rapidly in the past few years—so has the need for technologically competent citizens and skilled workers in computer and information science/engineering. Awareness of technology is becoming a de-facto core educational competency. The faculty of Wright State University have recognized the importance of technological awareness by recognizing two courses offered by the department as fulfilling general education competencies in the Wright State core.

Computation has become a *de facto* necessity for nearly all quantitative scholarly research. The department supports scholarly efforts throughout the institution (and region) by producing graduates and providing access to contributing state-of-the art computational experts. The impact of computation on the mission of universities is on the rise. Nation-wide, many academic institutions have embraced the mission critical nature of computation by investing significant resources into the founding of entire Colleges dedicated to Computing, Computing Sciences, and Computer/Information Science.

Faculty accomplishments and recognitions

The department currently consists of 34 full-time faculty members including 9 professors, 10 associate professors, 5 assistant professors, 3 senior lecturers, 2 lecturers and 5 instructors. The faculty includes 7 female faculty members (women are considered an underrepresented group in computer science & engineering). 26 of the faculty members in the department hold doctoral degrees from a variety of universities.

Faculty in computer science, computer engineering, and cyber security regularly receive funding to do research in areas of national and local need. Active projects funded by government and state agencies include:

- Instructional laboratories for cloud computing education (NSF)
- A federated semantic service platform for material sciences (DoD, AFRL)
- Fusion of multimodal video streams (AFRL)
- Developing large-scale language models (NSF)
- The Ohio consortium for bioinformatics (OBR)

Programs and areas of recognized excellence with supporting evidence

Bachelor of Science in Computer Science
The Bachelor of Science in Computer Science degree offers a curriculum in the study of the software aspects of computer systems including the study of algorithms and data structures, programming languages, software methodology and tools, data management and analysis.

Bachelor of Science in Computer Engineering
The Bachelor of Science in Computer Engineering degree offers a curriculum in the study of computer systems including the design, construction, and overall operations with a major focus on hardware. It includes the study of digital circuits, embedded systems programming languages and operating systems. The program provides a solid mathematics, basic science, and engineering science base that is common to all quality engineering programs

Bachelor of Arts in Computer Science
The Bachelor of Arts in Computer Science program prepares students for careers in computer systems analysis and design, programming, network administration and web site development. The Bachelor of Arts program allows for maximum flexibility of course study with fewer requirements in high level math and science. Our Graduates can pursue a wide array of careers and develop a broad understanding of multiple disciplines and the application of technology and computer in these fields.

Master of Science in Computer Science
The Department of Computer Science and Engineering offers a program of graduate study leading to a Master of Science degree in Computer Science. The program strengths include the unique blend of faculty expertise, well-equipped computer science laboratory facilities, and a balance of theory, practice, hardware, and software. Degree requirements concentrate on the areas of software system design and analysis. Courses for the program are offered in the late afternoon and evening hours to serve the educational needs of practicing computer professionals.

Master of Science in Computer Engineering
The Department of Computer Science and Engineering offers a program of graduate study leading to the Master of Science in Computer Engineering degree. The program strengths include the unique blend of faculty expertise, well-equipped computer engineering laboratory facilities, and a balance of theory, practice, hardware, and software. Degree requirements concentrate on the areas of computer system design and analysis. Courses for the program are offered in the late afternoon and evening hours to serve the educational needs of practicing computer professionals.

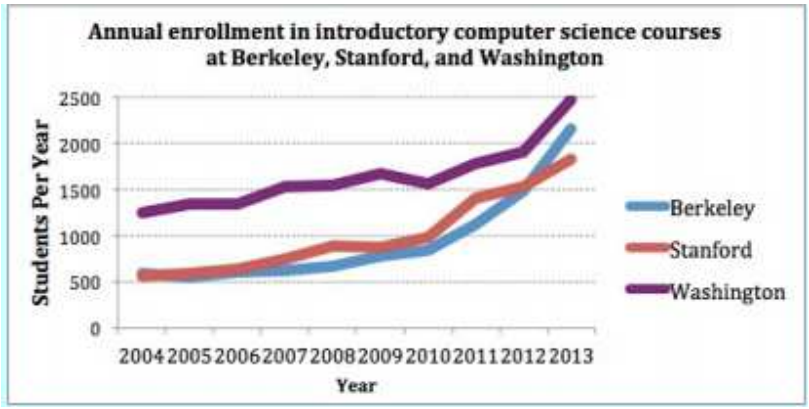
Master of Science in Cyber Security
The Master of Science in Cyber Security is designed for individuals who want to develop skills to identify and resolve Cyber Security threats. The degree is focused developing knowledge and skill applicable to protecting computer systems and computer networks. Program strengths include the unique blend of faculty expertise, the well-equipped computer engineering laboratory facilities, and the balance of theory, practice, hardware, and software.

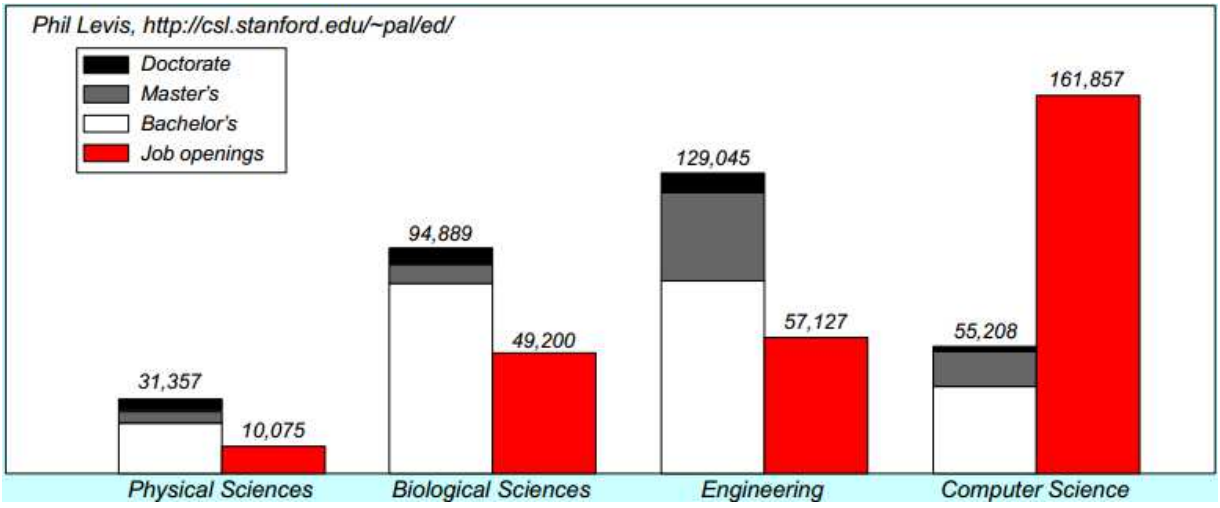
Doctor of Philosophy in Computer Science and Engineering
The Department of Computer Science and Engineering offers a program of graduate study leading to the

Doctor of Philosophy degree in Computer Science and Engineering. The Ph.D. degree is awarded in recognition of demonstrated, scholarly excellence in study and research that results in a significant contribution to the fields of computer science and/or computer engineering.

Capacity for growth of programs

The capacity crisis in Computer Science was a topic at the Jan 2014 NSF Priorities workshop. Dr. Eric Roberts (Stanford) reports IPEDS data, SIGCSE panels, and anecdotal evidence all indicate that CS enrollments are drastically increasing nationwide but that these increases are NOT keeping pace with the skyrocketing demand for computing professionals.





Universities nation-wide are failing to produce the number of computer scientists required to sustain economic growth. The table above [Phil Levis, Stanford University] demonstrates the gap between annual degrees granted (using IPEDS data 2008-2009) and employment data (Department of Labor, Occupational Outlook Handbook, 2010-2011).

The problem of the 2000s was insufficient student demand. The NSF Priorities presentation indicates that the problem of the 2010s will be insufficient university capacity. Computer science is poised to face a "success disaster" due to be unprepared to handle its overwhelming growth/need.

Historical evidence (from a similar situation in the 1980s) suggests that universities do not currently have the capacity to satisfy the growing demand. Workloads for faculty will increase substantially, some faculty members will leave for greener pastures, and replacement faculty will be increasingly difficult to find as students turn away from academic careers (or indeed, graduate studies altogether) to enter into immediately high-paying careers in the demand-rich job market.

Wright State University's department of Computer Science & Engineering will require substantial and continued investment in order to prepare for and meet the anticipated need BEFORE qualified faculty members become very difficult to recruit and retain.

At the graduate level, significant opportunity for growth is expected in the department's new online M.S. program in cyber security.

New program opportunities

The discipline of computer science/engineering is constantly redefining itself as technological needs and opportunities grow. The department is responsive to the changing needs of our students, the university, the state, and the nation. The department has recently created new programs in Cyber Security (MS-Cyber) and certificates in Cyber Security and in Big Data / Data Science.

The expected increase in computer science students nationwide will require new opportunities to meet more general needs for graduate students with less technical backgrounds. New programs under consideration should include not only new technical opportunities (such as a Master of Science in Data Analytics), but also less technical programs to meet the expended nation-wide demand for computer science professionals. These programs may include the developing of technical talent in individuals who have already demonstrated non-technical excellence, such as a Masters of Arts in Information Systems, "Fast track" BS programs for students possessing baccalaureate degrees in other disciplines, and similar opportunities.

Proposals to enhance programs (if desired)

- Promote excellence by improving teaching techniques to broaden the accessibility of the undergraduate program to a wider range of incoming student experience
- Promote excellence by increasing graduate and undergraduate participation in research by promoting the thesis option
- Continually adjusting admission standards as data suggests appropriate pre-admission metrics for predicting student success and ability to successfully complete the program

Assessment report, Fall 2014: Computer Science

Kathleen Timmerman, Travis Doom
Wright State University, Dayton, OH 45431-0001
Email: timmerman.16@wright.edu

Continuous assessment of student learning allows directed continuous improvement of the learning experience. Learning is multidimensional and requires multiple methods of collection in order to produce meaningful data. Direct methods of assessment measure student performance against some rubric of success. Indirect methods of assessment more often measure the student's (or observer's) perception of attainment. While both methods of assessment have their place, direct measures of assessment have been used for decades to provide a means for quality assurance. Historically, direct examinations such as the ACT and SAT have been used to measure the educational achievement of high-school students applying to college. Similarly, examinations such as the GRE, subject GRE, and Fundamentals of Engineering (FE) examination have been used to measure student educational achievement in University and to partially gauge professional competency.

Examinations of this sort provide validation against a set of external criteria that demonstrate that the *retained knowledge* of each student is *relevant* to the current national standard. Unfortunately, end-of-program examinations of this sort make poor tools for continuous program improvement. It is difficult, if not impossible, to provide a linkage between overall examination performance and specific actions or pedagogies employed in the educational process that led to greater or lesser success.

Continuous periodic direct measurements provide the best opportunity for measuring the performance effects of specific changes to programs, courses, and pedagogies. However, such data collection efforts are practically limited due to the sometimes massive effort required from administration, faculty, and students.

We use here an infrastructure to assess program effectiveness with the following goals:

1. The assessment provides continuous periodic direct measurements of retained relevant knowledge.
2. The assessment outcome is immediately valuable to the assessment participants (students and faculty) as well as the continuous improvement of the program.
3. The assessment is not unduly burdensome.

Assessment knowledge topics

The goal of assessment is to provide data to measure (or illustrate a need for) improvement. The definition of the assessment standards then set a target goal towards which a program continuously strives to better meet. Although program objectives differ significantly among institutions, certain knowledge and skills are expected of graduates of engineering programs. We believe that the standard towards which programs should strive in Engineering is best communicated not only by the accreditation agencies but also by the appropriate discipline-specific international

professional society. These societies maintain and regularly update the themes, knowledge areas, and professional practices expected of those entering their discipline.

For example, in computer science, the Joint Task Force on Computing Curricula between the Association for Computing Machinery (ACM) and IEEE-Computer Society provides regularly updated standards in curriculum, most recently in the volume Computer Science Curricula 2013 (CS2013) [1]. The CS2013 Body of Knowledge organizes the expectations of Computing graduates into 18 Knowledge Areas (KA) which are created, revised, and removed as the discipline changes over time (Figure 1, below). Each of these KAs is further specified as a set of Knowledge Units (Figure 2, below) each of which specifies a set of Knowledge Topics (Figure 3, below) expected at the time of graduation.

While acknowledging that every program has differing educational objectives, use of professional society standards provides metrics which can gauge the success of the program against a national model. Such metrics suggest an infrastructure for direct assessment that allows comparison against discipline-wide expectations and to allow reflection on the need, causes, and appropriateness of any major deviations from the widespread consensus proposed by the discipline's professional society.

AL	Algorithms and Complexity
AR	Architecture and Organization
CN	Computational Science
DS	Discrete Structures
GV	Graphics and Visualization
HC	Human Computer Interaction
IAS	Information Assurance
IM	Information Management
IS	Intelligent Systems
NC	Networking and Communication
OS	Operating Systems
PD	Parallel and Distributed Computing
PL	Programming Languages
SDF	Software Development Fundamentals
SE	Software Engineering
SF	System Fundamentals
SP	Social and Professional Practice

Figure 1: CS2013 Knowledge Areas [CS2013]

Algorithms and Complexity (AL)
AL/Basic Analysis
AL/Algorithmic Strategies
AL/Fundamental Data Structures and Algorithms
AL/Basic Automata Computability and Complexity

Figure 2: Sample Knowledge Units in the Algorithms and Complexity Knowledge Area [CS2013]

AL/Fundamental Data Structures and Algorithms

- Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max, and mode in a list, approximating the square root of a number, or finding the greatest common divisor
- Sequential and binary search algorithms
- Worst case quadratic sorting algorithms (selection, insertion)
- Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort)
- Hash tables, including strategies for avoiding and resolving collisions
- Binary search trees
- Common operations on binary search trees such as select min, max, insert, delete, iterate over tree
- Graphs and graph algorithms
- Representations of graphs (e.g., adjacency list, adjacency matrix)
- Depth- and breadth-first traversals
- Graphs and graph algorithms
- Shortest-path algorithms (Dijkstra's and Floyd's algorithms)
- Minimum spanning tree (Prim's and Kruskal's algorithms)
- Pattern matching and string/text algorithms (e.g., substring matching, regular expression matching, longest common subsequence algorithms)

Figure 3: Sample Knowledge Topics in the Algorithms and Complexity: Fundamental Data Structures and Algorithms Knowledge Unit [CS2013]

Continuous periodic direct measurements of retained relevant knowledge

We have worked with our program faculty to produce a mapping of which CS2013 Knowledge Topics are prerequisite to or developed in each of the core/mandatory courses in our computer science program. Our initial assessment framework is limited to mandatory "core" courses. Students will gain additional experience in many core knowledge topics in their elective coursework. However, the topics and amount of coverage will necessarily vary based upon the selected electives. Thus, initial observations are limited only to core/mandatory courses.

For each course, the faculty has indicated what knowledge topics are developed or assumed (prerequisite) in the semester-based course offerings. We propose that the current method to assess relevant retained knowledge is to perform a direct assessment of each knowledge topic not only in the course that develops that knowledge, but when possible, at the beginning of a subsequent course (or courses) that utilizes and builds on the topic.

Summative grading rubrics are, when possible, deployed at the start of the next course in the core course sequence (Figure 4, below). These assessment points allow better evaluation retention of expertise as measured prerequisite knowledge coming into each course. Assessment of prerequisite knowledge also allows assessment of differences among learning pathways, and are less subject to instructor- or course- related bias.

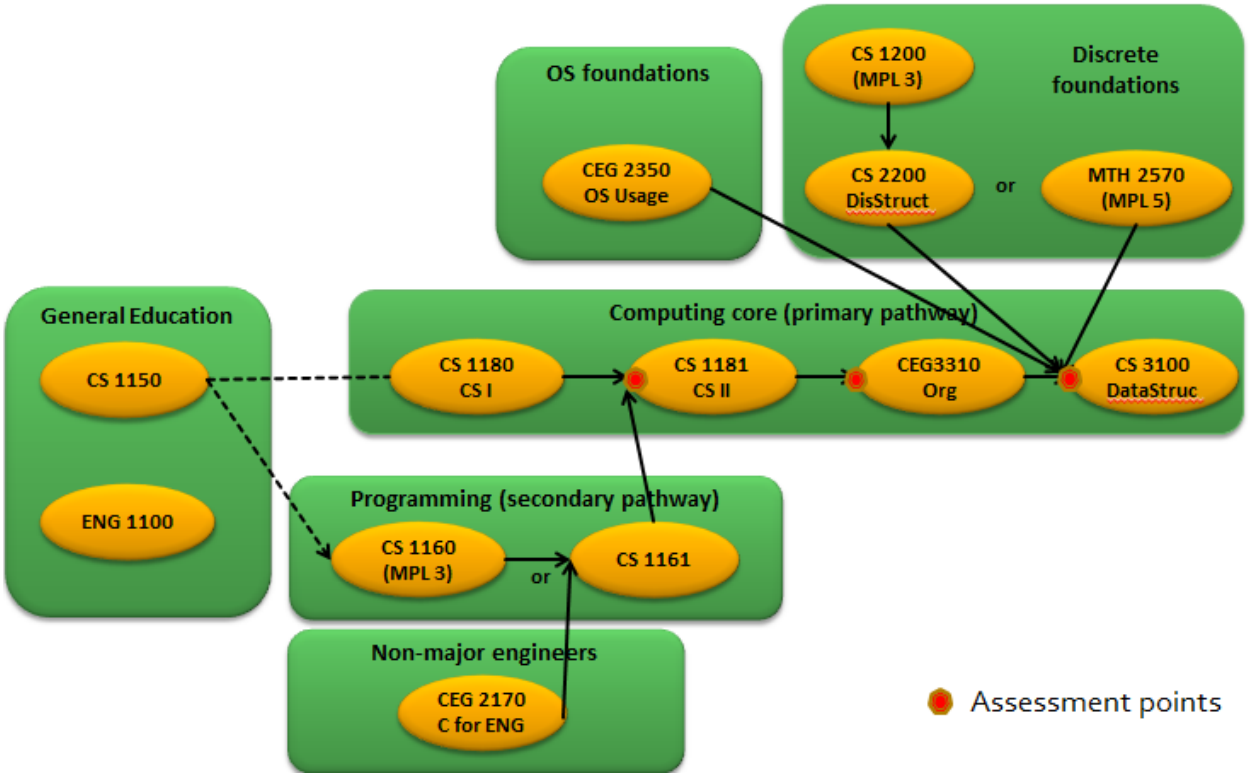


Figure 4: Assessment points in the first two years of the core curriculum. Additional assessment points exist in advanced core courses including Software Engineering, Operating Systems, and the Capstone Design Sequence. These assessment points are not illustrated. The assessments point for Capstone Design takes places at the end of that course sequence as there is no subsequent required course in the program.

Immediate value to participants

We encouraged that the assessments be required of all students entering a course but that the results not affect their upcoming grade in their current course. In our experience, students are very open about their level of mastery of concepts assessed in surveys of prerequisite knowledge. The feedback from these assessments is immediately useful to students as it calls up old ideas (helping them to be ready for new related knowledge) and can reduce anxiety regarding the sufficiency of their mastery of assumed prerequisite knowledge or identify specific areas where they can be coached to better prepare for succeed in a new course. As students find the feedback valuable to them personally, they are more likely to give significant and frank effort in the assessment process. As this effort is not associated with a course grade there is no need to proctor or use valuable classroom time on the assessment. The assessments are simply delivered as on-line standardized quizzes (Figure 5, below). Some instructors have chosen to use the first laboratory period (or other underutilized first week block of scheduled class time) to deliver the assessment surveys.

As a direct assessment of student preparedness, this data should be less biased than indirect assessments that ask students their opinion of their ability. Differences in self-expectation that may exist among students due to experience or demographic are removed. Thus, students/faculty get a more accurate measure of how well each student is prepared.

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.

```
int[] m = {2,3,4,5,6};
int n = 0;
int x = 0;
for (int val = 0; val < m.length; val++)
{
  if (val % 2 == 1)
  {
    n = n + m[val];
    x = x + 1;
  } // end-if
} // end-for
```

What is the most likely use for the code segment above?

A) Calculating the total sum of the values held in array m.
B) Calculating the average of the values held in array m.
C) Calculating the number of even values held in array m.
D) Calculating the average of odd values held in array m.
E) Calculating the number of values held in array m.

Figure 5: Sample Question to assess the Knowledge Topic AL/Fundamental Data Structures: Numerical Algorithms. This topic is developed in Computer Science I and built upon in Computer Science II. Thus, this question would appear in the assessment of prerequisite knowledge at the start of Computer Science II.

Equally important, the results of these perquisite surveys can be made immediately available to the faculty teaching the course in which the examination is held. If the faculty member sees weakness in prerequisite knowledge then they are able act to help address the problem immediately. The assessment can help identify individual students that might require additional help as well as identify potential systemic deficiencies introduced by previous poor instruction, variation in schedule due to weather/emergency, differing pathways for preparation (such as transfer courses), or the like. Based upon assessed performance, the faculty can tailor any necessary review of prerequisite topics appropriately to the needs of each term's student preparation.

Assessment overhead and administrative burden

Ease of assessment delivery allows the potential direct assessment of every student every term in every core course. As these assessments are delivered as on-line standardized examination, they require very little class time or faculty effort to administer. Each knowledge topic is mapped to relevant ABET engineering criteria 2000 CAC/EAC a-k criteria listed in ABET's Criterion 3: Student Outcomes [2,3]. This allows the data to be used by class or longitudinally by student to assess continuous improvement of the program overall against ABET Engineering criteria in a well-defined and straight-forward manner.

The most significant administrative burden is in the initial development, validation, and continuous improvement of the assessment questions. The initial burden of assessment development requires significant faculty involvement and may require multiple years of effort to construct assessment questions for every core course. The measurements for a knowledge area may be skewed by a set of poor assessment questions, thus continuous improvement of the questions in parallel with the improvement of curriculum remains an ongoing administrative effort.

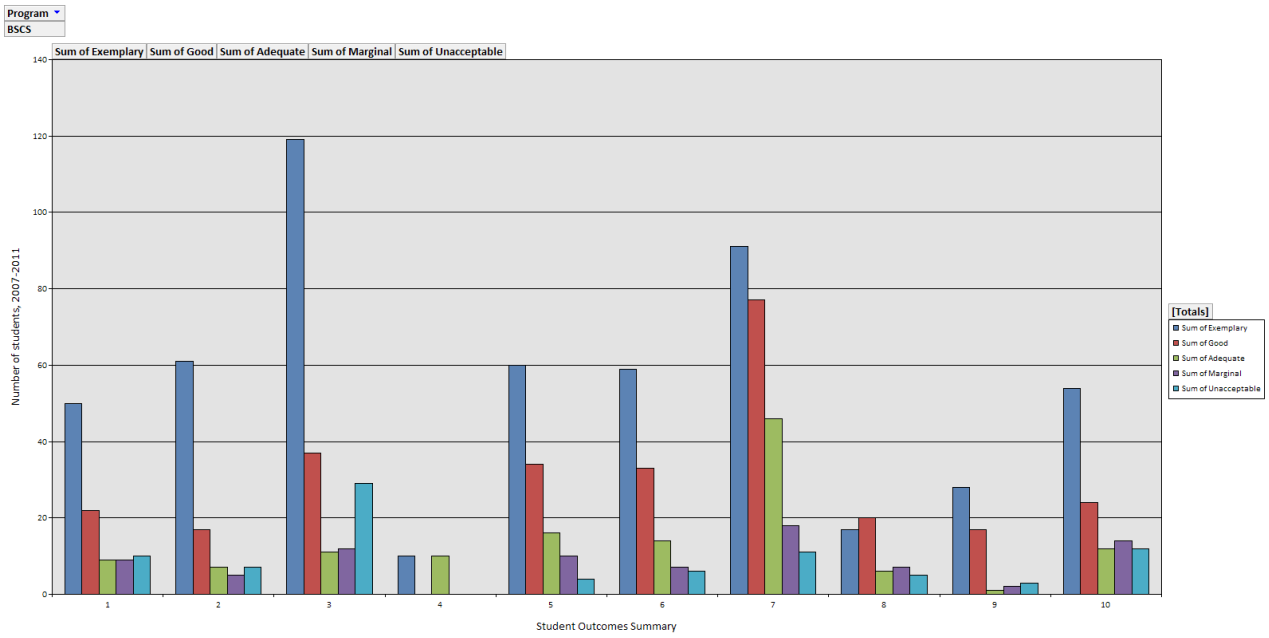


Figure 6: CSE Self-study results from previous assessment plan. The evaluation looks at Student Learning Outcomes and evaluates their coverage using a 5 point rubric measure for student performance: 1-Not at all, 2-To a limited extent, 3-To a moderate extent, 4-To a great extent, 5-To a very great extent

Assessment results used for Computer Science/Engineering self-study

Prior CSE assessment plans collected a rotating set of assignments so that each of the program's Student Learning Outcomes (SLOs) was assessed at least once every three years. The student's mastery of SLOs was assessed with an evaluation rubric ranging from 1 to 5 and summarized graphically using a device such as that show in Figure 6.

1. Not at all
2. To a limited extent
3. To a moderate extent
4. To a great extent
5. To a very great extent

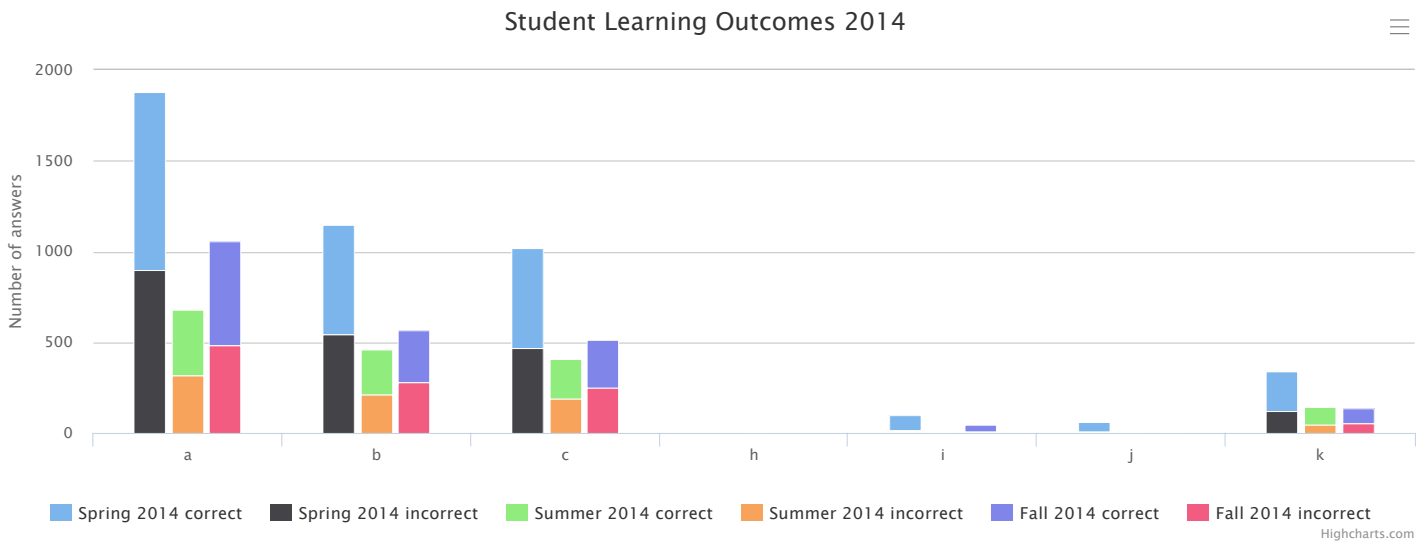
This style of visualization allowed for the identification of learning outcomes which had the most room for improvement. However, due to the timing consuming data collection (allowing for assessment of each SLO once every third year) it was difficult to use changes in measured outcomes in a meaningful way.

The new assessment system attempts to correct some of the issues that arose from the old system. First it uses a direct assessment system rather than an indirect summarized rubric measure. This reduces the subjective element that might cause inconsistency in determining how well expectations were met and also gives a numerical value that allows for meaningful measure of change significance. When there is a significant change, it is now possible to determine the cause of the change. Collecting data continuously (every SLO, every core course, every term) allows potential issues to be identified and addressed more rapidly.

What follows was created with the new assessment system as a web page.

New Self Assessment Report 2014

The richer data set of the new assessment infrastructure is maintained in a database allowing for specific drill down comparisons to compare different course preparation pathways, pedagogical styles, or any other variable of potential impact. For example, we can specifically address the question, is there a different in SLO achievement for students that take the two-semester intro-ductory computer science sequence for fully prepared incoming freshman (CS1180, CS1181) versus students that take the three course sequence for less prepared incoming students (CS1160, CS1161, CS1181).



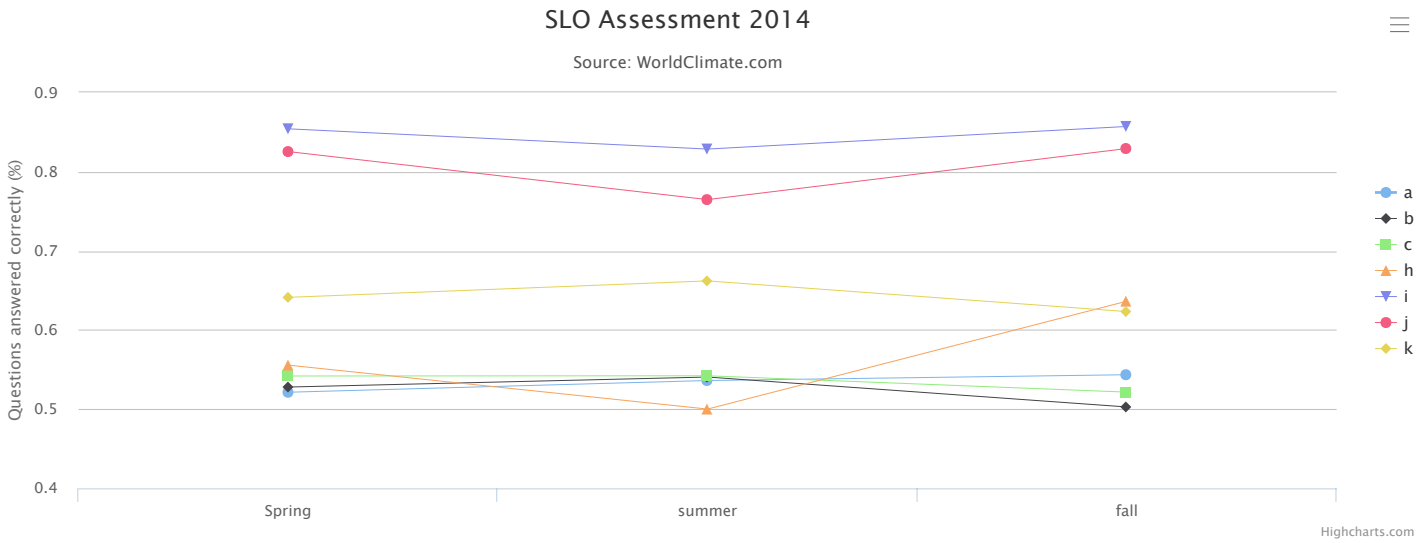
CAPTION: A comparison of how many assessment questions were answered correctly versus incorrectly during the Fall, Spring, and Summer terms of the 2014 school year. Each bar represents a student learning outcome for a given semester (number questions answered correctly to number answered incorrectly).

This chart shows the results of the assessment questions mapped to SLOs including the number of questions answered. The next chart shows similar data but presents it as percentage answered correctly based over term. By continuing to collect this data, we are able to watch for correlations such as impact of class size on student performance. Both charts are an example of the data that was collected for the self-study under the new system.

In addition to being able to look at the Student Learning Outcomes as a collective, this new as-sessment system makes it easier to break down the manner in which the data can be explored. It can be broken down based on courses, demographics, course preparedness, final grades, and stu-dent learning outcomes. Once broken down it can provide data providing feedback on the suc-cess or failure of changes to curriculum such as pathway options and teaching styles. The charts below are provided to show just some of the things that the new data collection system can ex-amine. Note that some of the visualizations are labeled "demonstration only" as data may take several semesters of assessment before delivering statistically significant results.

SLO combine table

Title	Correct	Incorrect
a	2801	2470
b	1770	1642
c	1613	1439
h	58	58
I	206	37
j	117	23
k	624	340



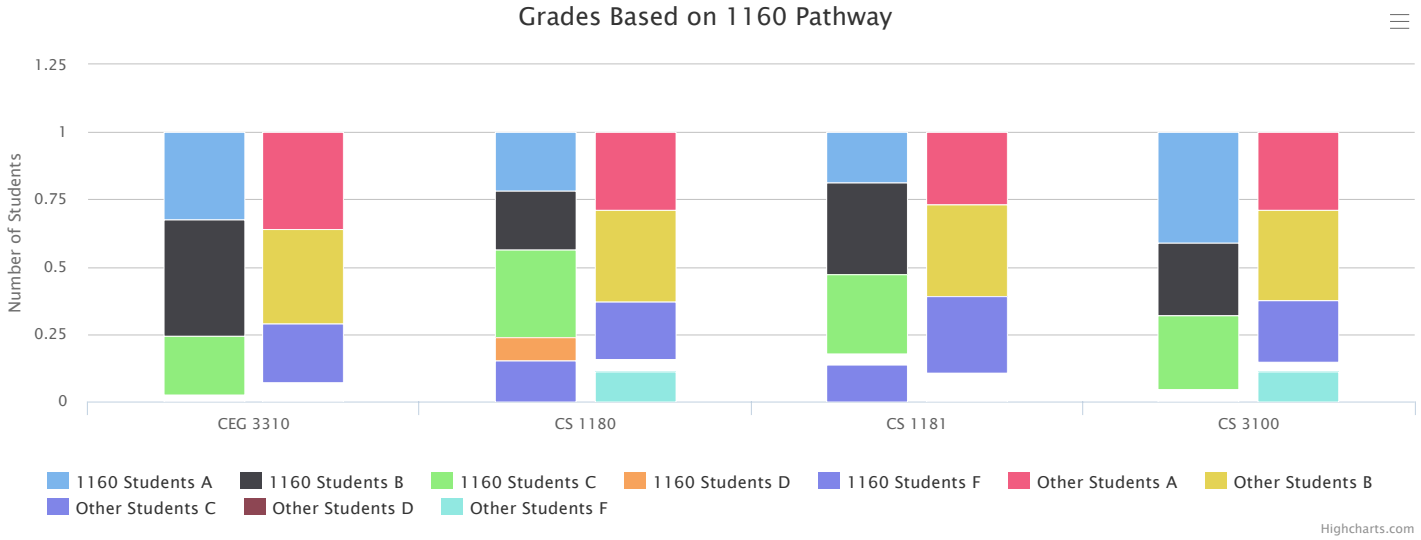
A different view of the comparison made in the above chart on how many assessment questions were answered correctly versus incorrectly during the Fall, Spring, and Summer terms of the 2014 school year. Here percentage answered correctly is used.

SLOs by Semesters 2014

Title	Spring Correct	Spring Incorrect	Summer Correct	Summer Incorrect	Fall Correct	Fall Incorrect
a	981	901	364	315	576	484
b	607	543	247	210	285	282
c	551	466	219	185	268	246
h	20	16	9	9	7	4
I	82	14	29	6	36	6
j	52	11	13	4	34	7
k	218	122	94	48	86	52

Different Pathways

Within our program, there are several different paths that a student could take (See Figure 4). It is assumed that the same knowledge is obtained from all the courses, but with the new assessment system, it can be easily monitored by performance on pre-assessment exams. In the next data sets the student results are broken down by pathways: those who took CS 1160/1, and those who did not (they either started in CS 1180 or transferred).



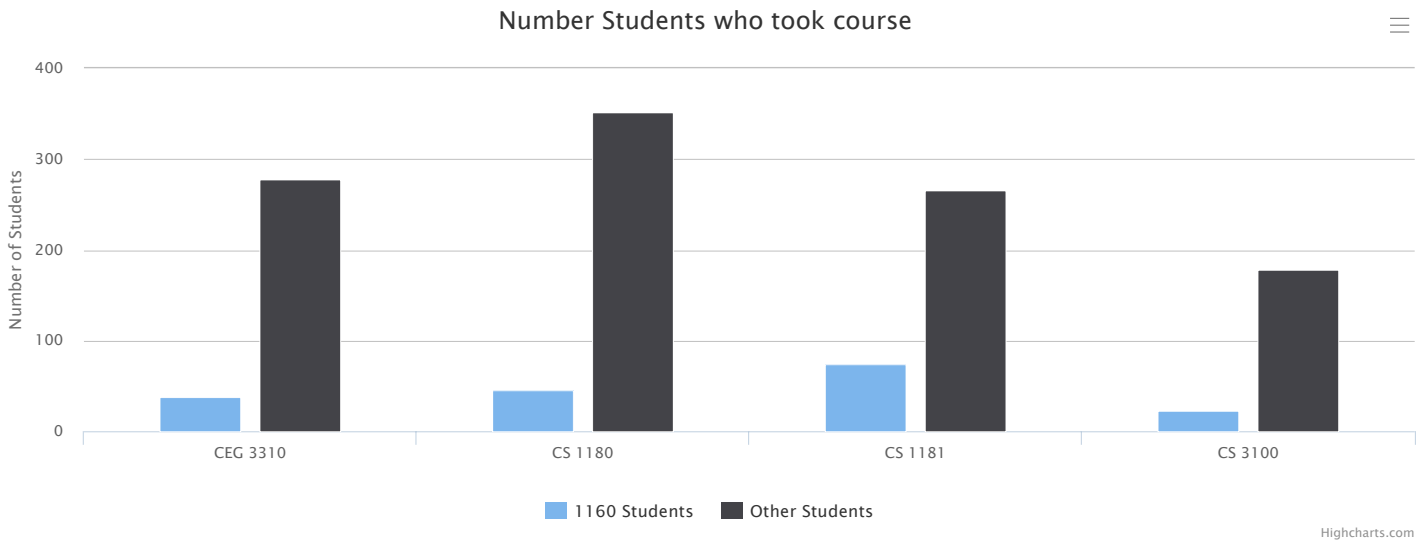
Students in 1160

Course	A	B	C	D	F
CEG 2350	20	23	18	12	16
CEG 3310	12	16	8	1	0
CEG 3320	6	5	3	0	3
CS 1150	10	18	2	5	0
CS 1160	83	91	42	24	62
CS 1161	50	36	22	1	6
CS 1180	10	10	15	4	7
CS 1181	14	25	22	3	10
CS 1200	44	66	73	29	32
CS 2200	8	32	47	13	10
CS 3100	9	6	6	0	1
MTH 2240	1	0	0	0	0
MTH 2280	3	2	1	3	4
MTH 2300	4	20	13	7	2
MTH 2310	1	8	3	3	3
MTH 2570	5	8	15	1	3

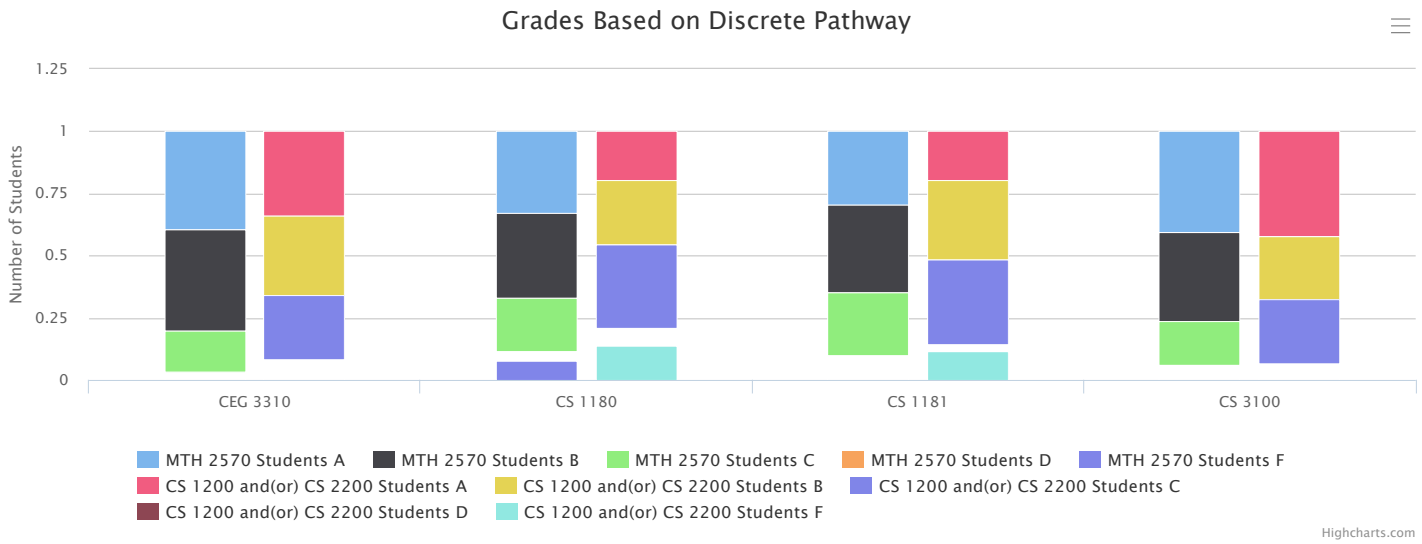
Students who started in CS 1180 or transfered

Course	A	B	C	D	F
CEG 2350	82	71	51	17	21
CEG 3310	100	97	60	8	12
CEG 3320	51	61	40	17	16
CS 1150	12	14	9	12	3
CS 1180	102	118	76	15	40
CS 1181	71	90	76	11	17
CS 1200	2	4	5	0	4
CS 2200	20	18	14	4	6
CS 3100	51	60	41	6	20
MTH 2240	2	6	1	0	0
MTH 2280	5	5	8	5	5
MTH 2300	27	48	63	22	16
MTH 2310	28	36	49	23	15
MTH 2570	67	76	98	17	19

The next chart looks simply at how many students took the course. This allows us to observe if there is a significant drop off of students from one of the pathways at some point through the core sequence.



The next data set is similar to the previous but it looks compares students who took CS 1200 and(or) CS 2200 versus those students who took MTH 2570. Both these pathways look at Discrete structures but the CS 1200/2200 sequence was designed for under prepared students.



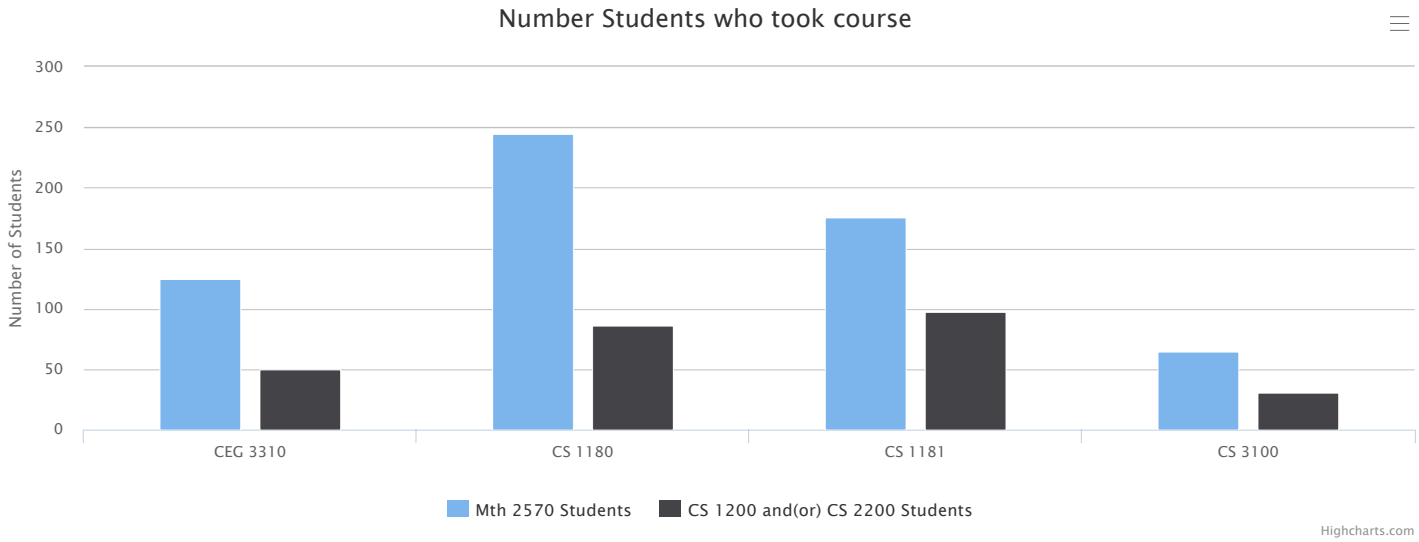
Students in MTH 2570

Course	A	B	C	D	F
CEG 2350	55	47	35	7	16
CEG 3310	49	51	21	0	4
CEG 3320	28	23	14	5	6
CS 1150	5	2	0	1	0
CS 1160	13	12	4	3	2
CS 1161	11	6	1	1	0
CS 1180	80	84	53	9	19
CS 1181	52	62	45	7	10
CS 1200	5	1	6	2	3
CS 2200	1	2	0	0	1
CS 3100	26	23	11	3	1
MTH 2240	2	3	1	0	0
MTH 2280	2	2	2	1	1
MTH 2300	20	37	31	15	7
MTH 2310	23	26	17	11	6
MTH 2570	72	84	113	18	22

Students in CS 1200 and(or) CS 2200

Course	A	B	C	D	F
CEG 2350	27	29	20	17	15
CEG 3310	17	16	13	2	2
CEG 3320	6	10	4	1	0
CS 1150	11	15	3	4	0
CS 1160	70	81	39	20	56
CS 1161	43	32	21	1	6
CS 1180	17	22	29	6	12
CS 1181	19	31	33	3	11
CS 1200	46	70	78	29	36
CS 2200	28	50	61	17	16
CS 3100	13	8	8	0	2
MTH 2280	3	4	2	4	3
MTH 2300	3	18	21	8	3
MTH 2310	0	7	5	6	5
MTH 2570	1	5	10	1	1

The next chart looks simply at how many students took the course. This allows us to observe if there is a significant drop off of students from one of the pathways at some point through the core sequence.



What follows is a breakdown of student answers by both knowledge topic and then question. While this is a little more cumbersome, it allows for specific problems to be identified.

Knowledge Topic Totals

Title	Correct	Incorrect
	538	436
Time and space trade-offs in algorithms	4	1
Asymptotic analysis of upper and average complexity bounds	3	2
Big O notation	0	5
Binary search trees	3	2
Boolean Statements	378	48
Brute-force algorithms	0	5
Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential	3	2
Depth- and breadth-first traversals	1	4
Differences among best, average, and worst case behaviors of an algorithm	4	1
Divide-and-conquer	2	3
Dynamic Programming	3	2
Empirical measurements of performance	2	3
Encoding	200	84
Exponents	478	56
Functions	674	148
Graphs and graph algorithms	2	3
Worst case quadratic sorting algorithms (selection, insertion)	3	2
Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort)	3	2
fields, methods, and constructors;	7	3
use;only;1	9	4
All	130	125
Arithmetic and geometric progressions	101	26
Arrays	112	388
Asymptotic analysis of upper and average complexity bounds	6	7
Basic modular arithmetic	44	83
Basic organization of the von Neumann machine	87	40
Basic syntax and semantics of a higher-level language	58	58

Binary Search Trees	57	70
Bits, bytes, and words	34	8
Comparison of algorithm efficiency	50	193
Complexity Classes	46	82
Compound types build from other types (e.g., records, unions, arrays, lists, functions)	82	34
Concept and properties	181	74
Conditional and iterative control structures	71	45
Counting arguments	37	90
Debugging strategies	32	84
Divide-and-conquer strategies	87	29
Documentation and program style	78	38
Doom1	7	6
Doom2	2	11
Events and event handlers	123	132
Exponents	65	13
Expressions and assignment	105	11
Fixed and floating-point systems	54	61
Functions	44	83
Functions and parameter passing	75	41
Fundamental design concepts and principles	218	25
Heap vs. Stack vs. Code segments	93	34
Induction	6	121
Iterative and recursive mathematical functions	80	36
Iterative and recursive traversal of data structure	72	44
Logical connectives	296	132
Method call	136	120
Numeric data representation	6	6
Numeric data representation and number bases	74	41
Numerical Algorithms	166	91
Object-Oriented design	103	24
Permutations and Combinations	180	54
Primative Types	135	120
Primitive types (e.g., numbers, Booleans)	94	23
Program correctness	47	81
Records	6	13
Recursive backtracking	107	20
Reference types	123	121
References and aliasing	101	142
Relations	48	79
Representation of records and arrays	33	95
Search	78	49
Sets	37	90
Simple linked structures	74	170
Stacks, queues, priority queues, sets & maps	63	181
Strategies for choosing the appropriate data structure	206	37
Strings	106	137
Subroutine call and return mechanisms	81	47
Subtyping	285	98
Sum and Product Rule	79	48
The concept and properties of algorithms	10	3
The concept of recursion	75	41
The pigeonhole principle	104	23
UNCLASSIFIED 2013 (Exponents)	177	33
Uninformed Search	88	39

Question Totals

Title	Correct	Incorrect

<address>Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
 	
boolean A = true;
boolean B = false;
boolean C = true;
boolean D = A || C;
boolean E = A & B || C;
boolean F = A || B & C;
 	
At the end of this segment of code, what is the value of the variable E? </address>
	217	39

<address>private List values; /** @return true if for every k in the range 0 < k < values.size(), * values.get(k) is less than values.get(k-1). *		

false   otherwise
 */
 public boolean isDecreasing() {
   boolean isDecr;
   /* statement 1 */
   for (int k = 1; k < values.size(); k++) {
     int prev = values.get(k-1).intValue();
     int curr = values.get(k).intValue();
     /* statement 2 */
   } // end for
   return isDecr;
 } // end method isDecreasing</address> <pre> </pre>   Consider the following replacements for statements 1 and 2   /* statement 1 */   /* statement 2 */   I.   isDecr = true;   if (curr >= prev) isDecr = false;   II. isDecr = false;   if (curr < prev) isDecr = true;   III. isDecr = true;   isDecr = isDecr && (curr < prev);   Which of the proposed replacements can be used so that isDecreasing() will work as intented?
	1	6

<address>private List<Integer> values;
 /** @return true   if for every k in the range 0 < k < values.size(),
 *   values.get(k) is less than values.get(k-1).
 *   false   otherwise
 */
 public boolean isDecreasing() {
   boolean isDecr;
   /* statement 1 */
   for (int k = 1; k < values.size(); k++) {
     int prev = values.get(k-1).intValue();
     int curr = values.get(k).intValue();
     /* statement 2 */
   } // end for
   return isDecr;
 } // end method isDecreasing</address> <pre></pre> Consider the following replacements for statements 1 and 2   /* statement 1 */   /* statement 2 */   I.   isDecr = true;   if (curr >= prev) isDecr = false;   II. isDecr = false;   if (curr < prev) isDecr = true;   III. isDecr = true;   isDecr = isDecr && (curr < prev);   Which of the proposed replacements can be used so that isDecreasing() will work as intented?
	59	190

Availnode is a pointer variable that points to the next available node in a singly linked list of available nodes. If p points to a node currently being accessed in a program, then the program fragment
 	
if p = nil then
    error
else
    p^.next:=availnode;
    availnode:=p
end
	2	8

 <address>Q.init();</address> <address>Q.enqueue(rootNodeOfTreeShownAbove);</address><address>while (! Q.isEmpty()) {</address><address> Q.dequeue(a);</address><address> if (!a.isNil()) {</address><address> a.print(); </address><address> Q.enqueue(a.left());</address><address> Q.enqueue(a.right()); </address><address> } // end-if</address><address>} // end-while</address><address></address> If the algorithm above is applied to the tree in the figure above, which of the following is the output? <address></address>	22	41
   Text Version of Tree: A (Root) -> B (Left Child of A) -> D (Left Child of B) -> E (Right Child of B) -> C (Right Child of A) -> F (Left Child of C) -> G (Right Child of C) Code: <pre> Q.init();</pre> <pre>Q.enqueue(rootNodeOfTreeShownAbove);</pre> <pre>while (! Q.isEmpty()) {</pre> <pre> Q.dequeue(a);</pre> <pre> if (!a.isNil()) {</pre> <pre> a.print();</pre> <pre> Q.enqueue(a.left());</pre> <pre> Q.enqueue(a.right());</pre> <pre> } // end-if</pre> <pre>} // end-while</pre> <address></address> If the algorithm above is applied to the tree in the figure (or text) above, which of the following is the output? <address></address>	35	120
 <pre>Q.init();</pre> <pre>Q.enqueue(rootNodeOfTreeShownAbove);</pre> <pre>while (! Q.isEmpty()) {</pre> <pre> Q.dequeue(a);</pre> <pre> if (!a.isNil()) {</pre> <pre> a.print(); </pre> <pre> Q.enqueue(a.left()); </pre> <pre> Q.enqueue(a.right()); </pre> <pre> } // end-if </pre> <pre>} // end-while</pre> <address> </address> If the algorithm above is applied to the tree in the figure above, which of the following is the output? <address> </address>	6	20
   Which of the following lists of nodes correspond to a postorder traversal of the binary tree in the figure shown?	78	49
   The edges of the digraph above are labelled by flow capacities (in, for example, gallons per hour). What is the maximum flow for this system from source A to sink K. In other words, what is the maximum flow (in, for examples, gallons per hour) possible from node A to node K.	40	87
<p>public static void main(String[] args) { A a = new B(); a.method(); } public class A { public void method(){ System.out.println("In Parent"); } } public class B extends A { @Override public void method(){ <span style="font-family: 'Courier'; font-size:	8	2

9.7pt;"> System.out.println("In child"); } } If the code above is run what will be printed?		
Consider a 16-bit data type such as java's integer data type. How many unique/distinct possible values could a variable of that type potentially be assigned? Choose the closest approximate answer.	106	51
Consider a 32-bit data type such as java's integer data type. How many unique/distinct possible values could a variable of that type potentially be assigned? Choose the closest approximate answer.	255	157
Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors. <address>int[] m = {2,3,4,5,6};</address><address>int n = 0;</address><address>int x = 0;</address> <address>for (int val = 0; val < m.length; val++) {</address><address> if (m[val] % 2 == 1) {</address><address> n = n + m[val];</address><address> x = x + 1;</address><address> } // end-if</address><address>} // end-for</address> Which of the following is the most likely intent for the code segment above? 	22	2
Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors. <address>int[] m = {2,3,4,5,6};</address><address>int n = 0;</address><address>int x = 0;</address> <address>for (int val = 0; val < m.length; val++) {</address><address> if (val % 2 == 1) {</address><address> n = n + m[val];</address><address> x = x + 1;</address><address> } // end-if</address><address>} // end-for</address> Which of the following is the most likely intent for the code segment above? 	40	17
Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors. <address>int[] m = {2,3,4,5,6};</address><address>int n = 0;</address><address>int x = 0;</address> <address>for (int val = 0; val < m.length; val++) {</address><address> if (val % 2 == 1) {</address><address> n = n + val;</address><address> x = x + 1;</address><address> } // end-if</address> <address>} // end-for</address> What is the most likely use for the code segment above? 	91	57
Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors. <address>int[] m = {2,3,4,5,6};</address><address>int n = 0;</address><address>int x = 0;</address> <address>for (int val = 0; val < m.length; val++) {</address><address> if (val % 2 == 1) {</address><address> n = n + val;</address><address> x = x + 1;</address><address>} </address> What is the most likely use for the code segment above? 	13	15
Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors. <address>public class Example {</address><address> private int x;</address><address> public void method (int y) {</address><address> y = y * 2;</address><address> x = y;</address><address> } // end method method</address><address> public int getValue() {</address><address> return x;</address><address> } // end method getValue</address><address>} // end class Example</address><address></address> The following code segment appears in a method in another class. <address>Example obj = new Example ();</address><address>int y = 10;</address><address>obj.method(y);</address><address>obj.method(y);</address><address>System.out.println(y + " " + obj.getValue());</address><address></address> What is printed as a result of executing the code segement?	136	120

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
 	
boolean A = true;
boolean B = false;
boolean C = true;
boolean D = A || C;
boolean E = A & B || C;
boolean F = A || B & C;
 	
At the end of this segment of code, what is the value of the variable D?
	149	8

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
 	
boolean A = true;
boolean B = false;
boolean C = true;
boolean D = A || C;
boolean E = A & B || C;
boolean F = A || B & C;
 	
At the end of this segment of code, what is the value of the variable E?
	137	20

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
 	
boolean A = true;
boolean B = false;
boolean C = true;
boolean D = A || C;
boolean E = A & B || C;
boolean F = A || B & C;
 	
At the end of this segment of code, what is the value of the variable F?
	135	22

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
 	
int[] arr;
arr = new int[5]; arr[0] = 0; for (int i=1; i < 5; i++) {
  arr[i] = arr[i-1] + (2 * i);
}
At the end of this segment of code, what is the value of arr[4]?
	82	34

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
 	
int a = 7;
int b = 2;
int c; c = a/b;
At the end of this segment of code, what is the value of the variable c?
	94	23

Consider the following segments of java code. Assume that there are no intentional syntax errors. <address>import javax.swing.*;</address><address>import java.awt.event.*;</address><address>public class Observer implements ActionListener {</address><address> public void buildGUI () {</address><address>   JFrame frame = new JFrame();   </address><address>   JButton button = new JButton("Click me");</address><address>   frame.add(button);   </address><address>   /* MISSING STATEMENT */   </address><address>   frame.setVisible(true);   </address><address> } // end method buildGUI</address><address>
</address><address> public void actionPerformed (ActionEvent event) {</address><address>   System.out.println("Click");</address><address> } // end method actionPerformed</address><address>} // end class Observer</address><address>
</address> Which of the
	123	132

following statements best replaces /* MISSING STATEMENT */ in the code segment above?		
The programming style in which design begins by specifying complex pieces and then dividing them into successively smaller pieces is know as:	218	25
You need to store a set of twenty objects in a computer program. The size of the collection is not going to change. Of the following options, which data structure would likely be the most efficient (in terms of execution time and memory usage) if used to store this collection?	206	37
A 0-2 binary tree is a rooted tree such that every node has either no children or two children. The height of a binary tree is the maximum number of edges on a path from the root to the leaf. Let n(h) be the minimum number of nodes in a 0-2 binary tree of height h, and let N(h) be the maximum number. For all h > 0, (n(h), N(h)) =	2	3
A 0-2 binary tree is defined to be a rooted tree such that evyer node has either no child or two children. What is the maximum height of a 0-2 binary tree that has 5 nodes?	77	50
A Graduate student says "I can write a super-merge sort which splits an array into 4 components (unlike a regular merge sort which splits an array into 2 components) and then recursively sorts the 4 subarrays and then merges them. Through similar to merge sort, it is faster and I call it SuperMerge4!." What is true?	2	3
A certain algorithm A has been shown to have running time O(N^{2.5}), where N is the size of the input. Which of the following is NOT true about algorithm A?	9	4
A complete binary tree of level 5 has how many nodes?	3	2
A die is tossed 7 times. What is the probability that all six faces appear at least once?	38	89
A double linked list contains references to both the predecessor and the successors elements and thus allows travel along the list in both directions. Consider a double linked list whose elements are declared as: <address>class Element {</address> <address> int value;</address><address> Element forward, backward;</address><address>} // end class Element </address> In this definition, forward and backwardd represent forward and backward links to adjacent elements in the list. Which of the following segments of code deletes the element pointed to by X from the doubly linked list, if it is assumed that X points to neither the first nor the last element of the list?	10	3
A heap H is used to implement a priority queue. The following values are inserted into the heap in this order: 44, 22, 55, 11, 44, 11, 33, 55. What value is stored at the right-most leaf of the bottom level of the heap?	2	11
A heap H is used to implement a priority queue. The following values are inserted into the heap in this order: 44, 22, 55, 11, 44, 11, 33, 55. What value is stored at the root of the heap?	7	6
A singly linked list is implemented in two arrays, value[] and link[], in which link[i] points to the successor of value[i]. If an element is not initially in the list assigned to value[j], then the program fragment <address> link[j] = link[i]; </address><address> link[i] = j;</address> is one that:	19	70
A singly linked list is implemented in two arrays, value[] and next[], in which next[i] points to the successor of value[i]. If value[j] is an element not currently in the list, then what does the following the program fragment accomplish? <address> next[j] = next[i];</address><address> next[i] = j;</address>	55	100
A time complexity function G was formulated for a program. What term asymptotically dominates G: G(x) = 9999999x - 0.0001x² + x * log (log(x)) + x²/(x-99) + 0.9^x	0	5
A two dimentional array A[1..row,1..col] is stored in memory begining at location S. Which of the following expressions points to the correct memory location for any arbitrary element A[i,j]?	3	10
A two dimentional array A[1..row,1..col] is stored in row-major order in memory begining at location S. Which of the following expressions points to the correct memory location for any arbitrary element A[i,j]?	30	85
An Internal hash table has 5 buckets, numbered 0, 1, 2, 3, 4. Keys are integers, and the hash function h(i) = I modulo 5 is used (i.e. h(i) = I % 5), with linear resolution of collisions (i.e. if bucket j(i) iis filled the buckets h(i) + 1, h(i) + 2, … are tried successively with all bucket numbers computed modulo 5). If elements with keys 13, 8, 24, 10, and 3 are inserted, in that order, into an initially blank hash table, then the content of the bucket numbered 2 is:	9	4
An algorithm that relies on recursion to break the problem into smaller more manageable pieces is a	10	0

Array ary is shown below <table class="MsoTableGrid" border="1" cellspacing="0" cellpadding="0" style="border-collapse: collapse; border: none;"> <tbody> <tr> <td width="31" valign="top" style="width: 23.4pt; border: solid windowtext 1.0pt; padding: 0in 5.4pt 0in 5.4pt;"> ‘h’ </td> <td width="30" valign="top" style="width: 22.5pt; border: solid windowtext 1.0pt; border-left: none; padding: 0in 5.4pt 0in 5.4pt;"> ‘i’ </td> <td width="24" valign="top" style="width: .25in; border: solid windowtext 1.0pt; border-left: none; padding: 0in 5.4pt 0in 5.4pt;"> ‘j’ </td> <td width="24" valign="top" style="width: .25in; border: solid windowtext 1.0pt; border-left: none; padding: 0in 5.4pt 0in 5.4pt;"> ‘k’ </td> </tr> </tbody> </table>
If 2 is passed to the method printArray below, what will the output be?
 	
public static void printArray(int index){
            if (index == 0){   System.out.print(ary[index]);   }
            else {
                    printArray(index -1);
                    System.out.print(ary[index]);
            }
}
	1	9

Assume array A is initialized with no syntax errors as: int[] A = new int[12]; Inside the variable A is a reference to a memory location.	10	0
Bob writes down a number between 1 and 1000. Mary must identify that number by asking “yes/no” questions of Bob. Mary knows Bob always tells the truth. If Mary uses an optimal binary search strategy, then she will determine the answer at the end of exactly how many questions in the worst case?	11	2
Bob writes down a number between 1 and 1000. Mary must identify that number by asking “yes/no” questions of Bob. Mary knows Bob always tells the truth. If Mary uses an sequential search strategy, then she will determine the answer at the end of exactly how many questions in the average case?	12	1
Consider N employee records to be stored in memory for on-line retrieval. Each employee record is uniquely identified by a social security number. Consider the following ways to store N records. (I) An array sorted by social security number (II) A linked list sorted by social security number (III) A linked list not sorted (IV) A balanced binary search tree with social security number as key For the structures I-IV, respectively, the average time for an efficient program to find an employee record, given the social security number as key, is which of the following?	11	2
Consider a floating-point number system used by a modern computer for solving large numerical problems. Let +		

_{fp} denote the floating-point addition in this system. Which of the following statements is true about this system?	60	67
Consider a routine in a C-like or java-like language that takes four integer arguments and returns an integer value. Which of the following approaches could be _reasonably_ used to _prove_ that the routine is correct? I. A complete test of all possible inputs. II. A mathematical proof of correctness. III. Testing a few well chosen test cases.	43	65
Consider a routine in a C-like or java-like language that takes four integer arguments and returns an integer value. Which of the following approaches could be reasonably used to prove that the routine is correct? I A complete test of all possible inputs. II A mathematical proof of correctness. III Testing a few well chosen test cases.	4	16
Consider a routine that merges together two unsorted linked lists into a single unsorted linked list. The unsorted linked lists are of size m and size n. What is the optimal average complexity for this routine?	6	7

Consider the Abstract Data Type (ADT) “set of integers” which is maintains an unsorted set of integers in which duplicates are allowed. Consider the following java-like code segments that are part of a proposed implementation of this ADT. <address>public static IntSet S = new IntSet();</address><address></address><address>final int MAX = 100;</address> <address></address><address>public class IntSet {</address><address>   int Last = -1;</address> <address>   int V[] = new int[MAX+1];
 } // end class IntSet</address><address></address><address> </address> The “add element” operation on an object S is implemented by storing the value of the element in S.V[S.Last + 1] and incrementing S.Last, unless Last = Max, in which case an error flag is raised.
The following code fragment mutates concrete IntSet objects.
  <address>public void P(IntSet S, int x)   {
   int k = 1;
   while (k <= S.Last) {
     if (S.V[k] = x) {
      S.V[k] := S.V[S.Last];
        S.Last := S.Last – 1
      } else {
        k = k + 1;</address><address>   } // end-if-else
   } // end-while
 } // end method P
 	
Which of the following abstract operations of “set of integers” does P implement?</address>
	9	4

Consider the Abstract Data Type (ADT) “set of integers” which is maintains an unsorted set of integers in which duplicates are allowed. Consider the following java-like code segments that are part of a proposed implementation of this ADT. <address>public static IntSet S = new IntSet();</address><address></address><address>final int MAX = 100;</address> <address></address><address>public class IntSet {</address><address> int Last = -1;</address> <address> int V[] = new int[MAX+1]; } // end class IntSet</address><address></address><address> </address> The “add element” operation on an object S is implemented by storing the value of the element in S.V[S.Last + 1] and incrementing S.Last, unless Last = Max, in which case an error flag is raised. Let A be a mathematical function that represents a “set of integers” such that, for each concrete object S of type IntSet, A(S) is the set of integers represented by S. A(S) can be written as:	4	9
Consider the Abstract Data Type (ADT) “set of integers” which is maintains an unsorted set of integers in which duplicates are allowed. Consider the following java-like code segments that are part of a proposed implementation of this ADT. <address>public static IntSet S = new IntSet();</address><address></address><address>final int MAX = 100;</address> <address></address><address>public class IntSet {</address><address> int Last = -1;</address> <address> int V[] = new int[MAX+1]; } // end class IntSet</address><address></address><address> </address> The “add element” operation on an object S is implemented by storing the value of the element in S.V[S.Last + 1] and incrementing S.Last, unless Last = Max, in which case an error flag is raised. What is the worse cast run time complexity of the "add element" operation for this proposed implementation?	10	3
Consider the code for a C function bump: <address>int bump (int x) {</address><address> int a;</address><address> a = x + 1;</address><address> return a; </address><address>} // end function bump</address> Some of the entries in the activation record for bump are placed on the stack by the function that calls bump; some are written by bump itself. Which are written by bump? I. Local variable a II. Arguement x III. Dynamic/Return link address 	12	1
Consider the code for a C function bump: <address>int bump (int x) {</address><address> int a;</address><address> a = x + 1;</address><address> return a; </address><address>} // end function bump</address> Some of the entries in the activation record for bump are placed on the stack by the function that calls bump; some are written by bump itself. Which are written by bump? I. Local variable a II. Argument x III. Dynamic/Return link address 	69	46
Consider the code for a C function bump: <address>int bump (int x) {</address><address> int y;</address><address> y = x + 1;</address><address> return y; </address><address>} // end function bump</address><address> </address> Where does the storage space for variable y exist in memory?? I. a Code Segment II. the Heap III. the Stack	93	34
Consider the following code segment, assume all variables are defined and initialized elsewhere. <address>for (int item = n; item >= 2; item--)</address><address> {</address><address> large = list[1];</address> <address> index = 1;</address><address> for (int i = 2; i <= item; i++)</address><address> {</address><address> if (list[i] > large)</address><address> {</address> <address> large = list[i];</address><address> index = i;</address><address> } // end if</address><address> list[index] = list[item];</address><address> list[item] = large;</address><address> } // end for i</address><address>} // end for item</address> The number of comparisions made by this algorithm is given by:	3	2
Consider the following code segment: <address>if (x % 2 == 0)</address><address> x = x - 1;</address> <address>if (x % 2 != 0)</address><address> x = x + 1;</address> If x is greater than 0 before the code segment is executed, which of the following states is (are) true regarding the final value of x after the code segment has executed? I. The final value of x is always even. II. The final value of x is equal to its initial value when x is initially even. III. The final value of x is equal to its initial value when x is initially odd.	130	125
Consider the following instance variables and incomplete method that will be used for computing a final quiz score. A final quiz score is computed by adding all the quiz scores and dropping the lowest score from the total. The method calculateFinalQuizScore performs the computation and updates the variable finalQuizScore. Assume that the instance variables are properly initialized. <address>private int[] quizScores;</address><address>private int finalQuizScore;</address><address></address><address>public void calculateFinalQuizScore() {</address><address> int		

total = 0;</address><address> int least = quizScores[0];</address><address> for (int k = 0; k < quizScores.length; k++) {</address><address> total = total + quizScores[k];</address><address> if (quizScores[k] < least) {</address><address> least = quizScores[k];</address><address> } // end-if</address><address> } // end-for</address><address></address><address> /* MISSING STATEMENT */</address><address>} // end method calculateFinalQuizScore</address><address></address> Which of the following can be used to replace /* MISSING STATEMENT */ so that calculateFinalQuizScore will work as intended?	181	74
Consider the following mathematical function: f(n) = f(n-1) + 2; f(0) = 17 How can a computer program be written to determine the value of f(n) for any value of n?	33	14
Consider the following mathematical function: f(n) = f(n-1) + 2 How can a computer program be written to determine the value of f(n) for any value of n?	47	22
Consider the following method: <address>Number max (Number x, Number y) {</address><address> if x.lessthan(y) {</address><address> return y;</address><address> }</address><address> return x;</address><address>} // end method max</address><address></address> In an object oriented programming language, which of the following is true? I. x and y can be of type Number II. x and y can of any type that is a subtype of Number III. x and y can be of any type that implements lessthan()	67	60

Consider the following program segment for finding the minimum value of an array: <address>public static double minValueInArray(double[] a) </address><address>{
   int i, j;
   j = 1;
   for (i = 2; i < a.length; i = i + 1) </address><address>   {
      if (a[i] < a[j]) </address><address>   {
         j = i;
      } // end-if
   } // end-for
   return a[j];
} // end method minValueInArray</address> Which of the following conditions is (are) true EACH time the condition of the IF statement is tested? I. 2 <= i <= a.length II. a[j] <= a[k] for all k such that 1<= k < i III. a[j] <= a[k] for all k such that 2 <= k <= i
	9	38

Consider the following program segment for finding the minimum value of an array: <address>public static double minValueInArray(double[] a) {
 int i, j;
 j = 1;
 for (i = 2; i < a.length; i = i + 1) {
 if (a[i] < a[j]) {
 j = i;
 } // end-if
 } // end-for
 return a[j];
 } // end method minValueInArray</address> Which of the following conditions is (are) true EACH time the condition of the IF statement is tested? I. 2 <= i <= N II. a[j] <= a[k] for all k such that 1<= k < i III. a[j] <= a[k] for all k such that 2 <= k <= i
	3	17

Consider the following program segment for finding the minimum value of an array: <address>public static double minValueInArray(double[] a) {
   int i, j;
   j = 1;
   for (i = 2; i < a.length; i = i + 1) {
      if (a[i] < a[j]) {
         j = i;
      } // end-if
   } // end-for
   return a[j];
 } // end method minValueInArray</address> Which of the following conditions is (are) true EACH time the condition of the IF statement is tested? I. 2 <= i <= a.length II. a[j] <= a[k] for all k such that 1<= k < i III. a[j] <= a[k] for all k such that 2 <= k <= i
	25	83

Consider the following program segment for finding the minimum value of an array: <pre>public static double minValueInArray(double[] a) {
 int i, j;
 j = 1;
 for (i = 2; i < a.length; i = i + 1) {
 if (a[i] < a[j]) {
 j = i;
 } // end-if
 } // end-for
 return a[j];
} // end method minValueInArray</pre> Which of the following conditions is (are) true EACH time the condition of the IF statement is tested? I. 2 <= i <= N II. a[j] <= a[k] for all k such that 1<= k < i III. a[j] <= a[k] for all k such that 2 <= k <= i
	15	54

Consider the following recursive definition of 4-Permutation (meaning permutations of 4 distinct objects): 1234 is a 4-Permutation If wxyz is a 4-Permutation then so are zyxw and xyzw According to this rule, the total number of permutations are:	6	121
Consider the following recursive function, written in a java-like language: <pre class="prettyprint">public static int fact(int n) {</pre> <pre class="prettyprint"> int result; </pre> <pre class="prettyprint"> result = n * fact(n - 1);</pre> <pre class="prettyprint"> return result;</pre> <pre class="prettyprint">}</pre> <pre class="prettyprint"> </pre> When this function is called as follows: fact(5), what value will be returned? 	75	41

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors. The operation (i % 2) returns 1 for odd values of i and 0 for even values of i. int sum = 0;
for (int i = 0; i < 10; i++) {   if (i % 2 == 0) {
    sum = sum + i;
  } else {
    sum = sum - i;
  } } At the end of this segment of code, what is the value of the variable sum?
	71	45

Consider the following segment of code in a java-like programming language. Assume that there are no syntax errors.
int a = 7;
int b = 4;
int c; c = 2 + 3 * a + b;
At the end of this segment of code, what is the value of the variable c?
	105	11

Consider the following segment of java code:
 	
int[] arr;
arr = new int[5]; arr[0] = 0; // The following line has an error:
for (int i=1; i <= 5; i++) {
 arr[i] = arr[i-1] + (2 * i);
}
Considering the error indicated by the comment, which of the following statements is correct:
	58	58

Consider the following segment of java-like code. Assume that there are no intentional syntax errors. <address>public abstract class Dog {</address><address> public abstract void speak();</address><address>} // end class Dog</address><address></address><address></address><address></address><address>public class LoudDog extends Dog {</address><address> public void speak() {</address><address> System.out.print("WOOF ");</address><address> } // end method speak</address><address>} // end class LoudDog</address><address></address><address></address><address></address><address>public class BabyLoudDog extends LoudDog {</address><address> public void speak () {</address><address> System.out.print("yip ");</address><address> } // end method speak</address><address>} // end class BabyLoudDog</address><address></address> Consider the following code segment. <address> LoudDog fido = new LoudDog();</address><address> BabyLoudDog rover = new BabyLoudDog();</address><address> fido.speak();</address><address> rover.speak();</address> What is printed as a result of executing the code segment?	218	38
Consider the following sequence of operations on an initially empty Queue Q. <address>Q.enqueue(1);</address><address>Q.enqueue(2);</address><address>x = Q.dequeue();</address><address>Q.enqueue(3);</address><address>Q.enqueue(4);</address><address>Q.enqueue(5);</address><address>y= Q.dequeue();</address><address>Q.enqueue(6);</address><address>z = Q.dequeue();</address><address></address><address></address><address>After these operations, what value is z?</address>	10	3
Consider the following sequence of operations on an initially empty stack S. <address>S.push(1);</address><address>S.push(2);</address><address>x = S.pop();</address><address>S.push(3);</address><address>S.push(4);</address><address>S.push(5);</address><address>y= S.pop();</address><address>S.push(6);</address><address>z = S.pop();</address><address></address><address>After these operations, what value is at the TOP of the stack?</address>	12	1
Consider the geometric progression: sum(n) = 1 + 1/2 + 1/4 + 1/8 + ... + 1/(2(n-1)) + 1/(2n) As n approaches infinity, what integer does sum(n) closely approach?	101	26
Consider the part of the two-dimentional integer grid bounded by the point A = (0,0) at the "southwest" corner and by point B = (n,n) at the "northeast" corner. How many different ways are there of walking from A to B on grid lines, always moving between any two grid points either east or north?	37	90
Consider the recursive routine, below: <address></address><address>int X (int n) {</address><address> if (n < 3) {</address><address> return 1;</address><address> } else {</address><address> return X(n-1) + X(n-3) + 1;</address><address> }</address><address>} // end method X</address> How many times is the function X called when X(X(5)) is evaluated?	5	8

Following is a recursive function for computing the sum of integers from 0 to N: <address>public static int sum (int n) {
   if (n == 0) {
      return 0;
   }
   // MISSING SEGMENT OF CODE
} // end method sum</address><address></address> In order to perform correctly, the missing segment of code would be best replaced by:
	107	20

For x >= 0, y >= 0, define A(x,y) by: A(0,y) = y + 1, A(x+1,0) = A(x,1), and A(x+1,y+1) = A(x, A(x+1, y)). Then, for all non-negative integers y, A(1,y) is:	44	83
Given the following sets : A ={0,1,2,3,4,5,6,7} B ={0,1,2,3,4,5,8,9} What is the Result of A - B :	7	3
Given the hash function: h = Data Item mod 60 If the chaining method is used, and if seven integer data items are stored in the hash table in the following order: 65 121 123 242 63 122 183 How many comparisions would it take to find integer data item 183?	2	3
How long, on average, does it take to find a item in an unsorted list of size 10? Select the closest value below.	88	39
How many Bytes (groupings of eight binary digits) end in the sequence 000?	79	48
How many comparisons are required to sort an array of length 5 if a straight selection sort is used and the array is already sorted in the opposite order?	0	5
How many different values can be represented using 8 bits?	91	26
How many distinct values can be encoded for storage using one byte (8 binary digits) of memory per encoding?	104	23
If you are not careful in your choice of hash functions, it is possible to have collisions in which the search time goes to the order of O(?) in a search table of size n:	4	1
In Java, variables that store reference types can be: I. used to implement call by reference II. potentially subject to aliasing III. used to implement call by value	81	74
In Java/C-like languages, variables that store reference types are: I. Call by reference II. Potentially subject to aliasing II. Call by value	31	32
In Java/C-like languages, variables that store reference types are: I. Call by reference II. Potentially subject to aliasing III. Call by value	11	15
In a survey of what types of courses students were enrolled in one term, it was found that: 520 students took a CS course 416 students took a Math course 320 students took a CEG course 152 students took CS and Math 96 students too CS and CEG 124 took Math and CEG 60 took CS, Math, and CEG Using Venn Diagrams and the Counting Principle (inclusion/exclusion), compute how many students are taking exactly ONE of the three types of courses (no more, no less).	37	90
In how many unique ways can the letters in the word "ABOUT" be arranged?	89	28
In most contemporary C-like programming languages, a String is stored as: I. A primative data type II. A reference data type III. An array of characters	106	137
In order to traverse a binary tree using the inorder traversal method, the statements in the code segment shown below should be: <address>public class Node</address><address>{</address><address> Datatype data;</address><address> Node left, right;</address><address>} // end class Node</address><address></address><address>public void method traverse (Node current)</address><address>{</address><address> if (Node.data != nil) {</address><address> // WHAT CODE GOES HERE?</address><address> } // end-if</address><address>} // end method traverse</address>	1	4

In the following form of binary search one can determine whether some particular integer x is present in a sorted (in ascending order) array a[L..R] using the following steps in the order shown. Let M= (L + R) / 2 If x = a[M], then terminate (success). If x < a[M] and M > L, use the same algorithm on a[L .. (M-1)] If x > a[M] and M < R, use the same algorithm on a[(M+1) .. R]. Otherwise terminate (failure). Which of the following statements is (are) true about binary search in a[1..N], where N is a large positive integer? (I) If x is present in a[1..N], then x is always found within O(log N) comparisons. (II) If x is not present in a[1..N], then failure is always reached within O(log N) comparisons (III) Searching for two different values of x, neither which is present in a[1..N], always takes the same number of steps to determine failure.	7	3

In the following program segment, assume that when the execution of the segment begins, that m <= n AND that a[m-1] <= v <= a[n+1]. <address>int i = m - 1;
 int j = n + 1;
 while (i < j) {
   do {
      i++;
   } while (a[i] < v);
   do {
      j--;
   } while (a[j] > v);
   if (i<j) {
      temp = a[i];
      a[i] = a[j];
      a[j] = temp;
   } // end-if
 } // end-while</address> <pre></pre> <pre>For large values of n-m, which of the following best approximates the sum of the number of times the assignments i++ and j-- are executed? </pre>
	50	193

Inheritance makes it easier to:	3	7
Inherited attributes are	0	10
Let A be a sorted array of n = 10 elements. Which of the following denotes the average successful time for finding an arbitrary element x in A using a BINARY SEARCH? Assume that only one comparision is required to determine whether the target is equal to, less than, or greater than A[i].	57	70
Some programming languages pass parameters to methods/functions by value, while others pass parameters by reference. If a function is passed two parameters by value, which of the following statements is most correct:	75	41
Suppose you are asked to write a function that finds the largest value in an array of integers. Which of the following statements is true.	72	44
TBD	6	13
The Boolean expression: NOT(A OR A AND B) is equivalent to:	36	91
The average time required to perform a successful sequential search for an element in an array of size n is given by:	78	49
The basic organizational units of a modern computer are:	87	40
The binary relation on the integers defined by R = { (x,y) : | y - x | <= 1 } has which of the following properties? I. Reflexivity II. Symmetry III. Transitivity	48	79
The key attribute(s) that a problem just have in order for dynamic programming to be applicable are: i) linear consumption ii) optimal substructure iii) overlapping subproblems	3	2
The number of 1's in the binary representation of 13 * 16³ + 11* 16² + 9 * 16 + 3 is which of the following?	82	45
What is 2⁰?	379	32
What is 2¹⁰x 2¹⁰? Please choose the closest answer.	341	70
What is the powerset of {a,b,c}?	7	3
What will the following program segment accomplish: Scanner inData = new Scanner(new File("infile.txt")); PrintWriter outData = new PrintWriter("outfile.txt"); while(inData.hasNext()){ outData.printLine(inData.nextLine()); } inData.close(); outData.close();	10	0
Which of the following best seems a reasonable use for modular arithmetic?	44	83
Which of the following is not in important element of good programming style:	78	38
Which of the following lines of code will delete two successive nodes of a singly linked linear list (WITH MORE THAN 2 NODES)? Assume this code is in the main program, not a subprocedure.	7	3
Which of the following shows the correct relationship among some of the more common computing times for algorithms?	3	2
Which of the following shows the correct relationship amoung some of the more common computing times for algorithms?	46	82
Which of the following sorting algorithms can be characterized as using a divide-and-conquer strategy.	87	29
Which of the following sorting algorithms has average-case and worst-case running times of O(n * log(n))?	3	2
Which of the following sorting algorithms yield approximately the same worst-case and average-case running time behavior in O(n * log n)?	4	1
Which sort will operating in quadradic time relative to the number of elements in the array (on the average)?	3	2
Which statement about objects is true?	103	24
Your Java program seems to compile correctly. When run, however, it runs forever with no output. Which of the following debugging strategies is the most likely to help you find the error?	32	84
f(x) = kx²- 4 If f(4) = 4, then what is f(7)?	308	103
f(x) = x²- 4 What is f(3)?	366	45
if class a is derived from class b and class b is derived from class c, polymorphism allows	8	2

<pre>public class Point {</pre> <pre> int x;
 int y;</pre> <pre></pre> <pre> public Point (int x, int y) {</pre> <pre> this.x = x;</pre> <pre> this.y = y;</pre> <pre> } // end constructor</pre> <pre></pre> <pre> public static void main(String[] args) {</pre> <pre> Point p1 = new Point(1,2);
 Point p2 = new Point(3,4);
 p2 = p1;
 p1.x = 5;
 System.out.println(p2.x);</pre> <pre> } // end method main</pre> <pre></pre> <pre>} // end class Point</pre> <pre></pre> <pre>Consider the code segment above? What value is output for p2.x?</pre>
	101	142

A algorithm that tests every possible solution until the solution is found is a	8	2
A component that speeds up computer processes and stores frequently used data is the	10	0
A data structure where elements can be added or removed at either end but not in the middle is called a …	3	7
A simple (undirected or directed) graph is one in which there are no self loops and no multiple edges. An undirected graph is acyclic if it has no cycles. What is the maximum possible number of edges in an n-node, simple, acyclic, undirected graph?	18	5
An Exception is thrown due to:	5	6
Consider a data type whose elements are integers and whose operations are INSERT, DELETE, and FINDCLOSEST, with FINDCLOSEST(y) defined to be some element x in the current set such that lx - yl <= lx_i- yl for all x_i in the current set. Let T = max (T_{INSERT},T_{DELETE},T_{FINDCLOSEST}) where T_{OP} denotes the worst-case time complexity for the given operation OP. Which of the following data structures would be best to use in order to minimize T?	6	7
How many explicit constructors can a class have?	7	3
In a height-balanced binary search tree, the heights of the left and right descendents of any node differ by at most 1. Which of the following are true of such a tree? (I) Worst-case search time is logarithmic in the number of nodes. (II) Average-case search time is logarithmic in the number of nodes. (III) Best-case search time is proportional to the height of the tree. (IV) The height of the tree is logarithmic in the number of nodes.	8	5
Of the following sorting algorithms, which has a running time that is LEAST dependent on the initial ordering of the input?	11	12
The intersection of the two regular languages below: L₁= (a+b)*a and L₂= b(a+b)* is given by:	2	8
The proper operator precedence groupings, from most binding to least binding, are:	4	6
What purpose does class construct serve?	8	2
Which of the follow is true?	10	0
Which of the following sorting algorithms has average-case and worst-case running times of O(n log n)?	7	6
Which sorting algorithm considers the elements one at a time, inserting each element in its suitable place among those already considered (keeping them sorted)?	9	1
Which statement is true? (members = fields and methods)	7	3
With inheritance, the derived class	10	0
_____ is when a computers system is in a constant state of paging.	6	4

Flexibility with the New Assessment System

It is important to note that anything that is based on pre-assessment grade can easily be linked to student learning outcomes and program educational objectives since all the questions are mapped to what they test. Similarly, anything that shows student learning outcomes or program educational objectives can be broken down into specific knowledge topics, questions, and students who answered those questions. Since all the results are linked back to a specific student, all the questions can be broken down by demographics, year began, pathways, or even previous performance in a course or on a pre-assessment quiz. This makes the items displayed in this paper easy to mix and match. Also note that while charts were largely used in this paper for ease of viewing, all the values in the charts have hard numbers backing them up. Statistical tests can be run on areas of interest to determine if a change really occurred before any action is taken. Additionally, the results can be presented as exact sample size or as percentages.

Conclusion

This assessment infrastructure allows for an assessment of retained knowledge, topic by topic, for each individual student, course, and term. When collected with appropriate demographic information, these assessments allow the differential measurements of knowledge retention under any number of pedagogical variables. The success of new instructional styles, laboratory techniques, or technologies for developing knowledge can be assessed against different approaches.

Every contemporary engineering discipline has a professional society that helps identify the core concepts of the discipline. Indeed, most engineering disciplines have standardized examinations of some sort that are used to demonstrate student proficiency for licensure or graduate studies. Questions of this sort can be used at the start of core courses or time points to assess student knowledge of prerequisite topics developed earlier in any program of study. These assessments can be delivered as online questions to minimize cost and maximize participation. When collected with appropriate demographic information, this rich set of data can guide program improvement more effectively than many existing program assessment plans. Although we present this infrastructure in the context of Computer Science, we believe that the approach can be applied to implement an infrastructure for effective assessment program for any engineering discipline.

Biblography

[1] ACM/IEEE-CS Joint Task Force. Computer Science Curricula 2013 (CS2013). http://ai.stan¬ford.edu¬/users-/sahami¬/CS¬2013/. Strawman draft, Feb 2012.

[2] ABET. Computing Accreditation Commission (CAC) Criteria. http://www.abet.org/accreditation-criteria-policies-documents/

[3] ABET. Engineering Accreditation Commission (EAC) Criteria. http://www.abet.org/accreditation-criteria-policies-documents/

Computing Accreditation Commission

Final Statement of Accreditation

to

WRIGHT STATE UNIVERSITY

Fairborn, OH

2011 – 2012 Accreditation Cycle

FINAL STATEMENT

This is a confidential statement from the Computing Accreditation Commission to the institution. It is intended for internal use only and is not for release except as allowed by policies of ABET.

I. INTRODUCTION

Founded in 1964 and granted full university status in 1967, Wright State University is a comprehensive, state-supported university located in Fairborn, Ohio, with a satellite campus in Grand Lake St. Marys, Ohio. Wright State has nearly 20,000 students of which over 15,000 are full-time students and approximately 4000 are graduate and professional students. The university has 760 full-time faculty members supporting 19 associate, 91 undergraduate, and 76 graduate and professional degree programs.

The following programs at the institution were evaluated during the 2011-12 cycle for possible accreditation under the CAC/ABET "Criteria for Accrediting Computing Programs" (*Criteria*) dated October 30, 2010:

- BS Degree in Computer Science, evaluated under the General Criteria and the Computer Science Program Criteria. The BS program in Computer Science was previously evaluated in 2005. As a result of that accreditation action, the institution was required to submit an interim report in July 2007. As a result of the evaluation of the interim report, the institution was required to submit an additional interim report in July 2009. As a result of this second evaluation of the interim report, the institution would have been required to submit an additional Interim Report in July 2011; however, since the cycle for the evaluation of that interim report coincided with its NGR cycle, no report was required.

- BS in Business Degree in Management Information Systems, evaluated under the General Criteria and the Information Systems Program Criteria. The BS program in Management Information Science was previously evaluated in 2005. As a result of that accreditation action, the institution was required to submit an interim report in 2007. As a result of the evaluation of the interim report, accreditation was extended to 2012.

The programs listed above were evaluated under the CAC/ABET "Criteria for Accrediting Computing Programs" (*Criteria*) dated October 30, 2010 by the peer review team shown below.

- Team Chair: David Gibson, United States Air Force Academy
- Program Evaluator: David Bover, Western Washington University
- Program Evaluator: Subhasish Dasgupta, George Washington University
- Editor One: Judith Solano, University of North Florida
- Editor Two: Harold Grossman, Clemson University

Please note that program accreditation decisions are made solely by the respective Commissions of ABET. Reference to the professional affiliations of the volunteer peer evaluators in no way constitutes or implies endorsement or recommendation of the programs by the listed professional affiliations.

II. REPORT OF FINDINGS

The *Criteria* is composed of the General Criteria and Program Criteria. Each criterion provides the underlying principles that each program must meet. A program must meet both the General Criteria and all applicable Program Criteria to be accredited.

This section contains the findings from the time of the visit. It also includes an evaluation of any information provided by the program during the due process response. CAC considers the following comments to relate directly to its accreditation actions.

A program's accreditation action will be based upon the findings summarized in this statement. Actions will depend on the program's range of compliance or non-compliance with the criteria. This can be determined from the following terminology:

- Deficiency: A deficiency indicates that a criterion, policy, or procedure is not satisfied. Therefore, the program is not in compliance with the criteria.

- Weakness: A weakness indicates that a program lacks the strength of compliance with a criterion, policy, or procedure to ensure that the quality of the program will not be compromised. Therefore, remedial action is required to strengthen compliance with the criterion, policy, or procedure prior to the next evaluation.

- Concern: A concern indicates that a program currently satisfies a criterion, policy, or procedure; however, the potential exists for the situation to change such that the criterion, policy, or procedure may not be satisfied.

- Observation: An observation is a comment or suggestion that does not relate directly to the accreditation action but is offered to assist the institution in its continuing efforts to improve its programs.

Computer Science Program

The B.S. in Computer Science is offered by the Department of Computer Science and Engineering in the College of Engineering and Computer Science. Students majoring in computer science may choose from five options: Bioinformatics, Business, Computational Science, General, and Visualization. The department also offers MS and PhD degrees in computer science as well as BS, MS, and PhD degrees in Computer Engineering. The BS in Computer Engineering program is accredited by the Engineering Accreditation Commission of ABET. The department has 26 full-time faculty members supporting both the Computer Science and Computer Engineering programs. There are 306 students enrolled in the CS major. All programs are properly differentiated in university publications.

Program Strength

The Department of Computer Science and Engineering has a faculty member with exceptional experience and skill in technical writing. As a result, students in the computer science program receive instruction in technical writing at a level of quality and rigor significantly above that normally found in such programs. Interviews with students confirmed the extraordinary value placed on this course. As a result, students from the program are exceptionally well prepared in technical writing.

Status of Shortcomings from the Previous Review

Weakness

1. Criterion I. Objectives and Assessments. The following factors contribute to this weakness:
   a. (Standard I-3) There is a lack of documentation to show how the data collected are used to identify opportunities for program improvement.
   b. (Standard I-6) There is a lack of documentation to show how program improvements are related to program assessment findings.

   Status: This weakness has been resolved.

Findings from the Current Review

Weaknesses

1. Criterion 3, Student Outcomes. The program has adopted the characteristics listed in the criterion as its outcomes. Outcome (b) requires that the program must enable students to attain, by the time of graduation, *an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution*. The team observed that only one course includes work in problem analysis, providing minimal opportunity to develop problem analysis and requirements specification skills. As a result, graduates of the program may lack developed skills in problem analysis and requirements specification.

Due-process response: Since the visit, the program has increased problem analysis requirements on two assignments in the required Computer Science III course (CS 242). The program has also increased emphasis on problem analysis in the required senior-level Introduction to Software Engineering course (CEG 460) with two additional lessons on analysis, incorporating use of a visual modeling tool, and addition of more analysis work in homework assignments and on the course project. These changes were put into place during the Winter 2012 quarter. The program has also initiated a curriculum change that will require a two-semester capstone sequence, Team Projects I and II (CS 4980 and CS 4981) required for all Computer Science majors in the graduating class of 2016 and perhaps as early as the graduating class of 2013. This proposed change is expected to receive formal approval by the university's Faculty Senate in May or June 2012.

Due-process evaluation: The weakness remains unresolved. The program has taken appropriate steps to increase problem analysis and requirements specification in the current academic year. The program will change to a semester-based curriculum next fall. If the proposal for requiring the 2-semester capstone sequence is formally approved and changes made to Computer Science III this past Winter quarter are carried into the new semester-based curriculum as planned, future graduates of the program will have appropriate experiences in problem analysis and requirements specification. However, until the new capstone sequence is approved, this weakness remains unresolved.

Post due-process response: The program has reported formal WSU University Curriculum Committee approval of the requirement for all computer science majors, starting with the graduating class of 2014, to take the two-semester capstone sequence, Team Projects I and II (CEG 4980 and CEG 4981). The program also provided updated syllabi for these courses showing course outcomes and content addressing problem analysis and requirements specification.

Post due-process evaluation: The weakness is now cited as a concern. The program has taken appropriate steps to increase the coverage of problem analysis and requirements specification content in courses required for the computer science major. Coverage of these topics has been expanded and taught in the required Computer Science III (CS 242) and Software Engineering (CEG 460) courses. The recently approved new courses, Team Projects I and II (CEG 4980 and CEG 4981), should provide for even more thorough coverage of the topics when the courses are regularly taught. This level of coverage of problem analysis and requirements specification needs to be maintained throughout the period of accreditation.

2. Criterion 4, Continuous Improvement. The Continuous Improvement criterion requires that the program must *regularly use appropriate, documented processes for assessing and evaluating the extent to which both the program educational objectives and the student outcomes are being attained.* The results of these evaluations must be systematically utilized as input for the continuous improvement of the program. The team observed that although the department has implemented a process for collection of assessment data in a database, they are not yet able to demonstrate systematic utilization of that data for continuous

improvement. As a result, the program may miss opportunities for improvement based on program assessment.

Due-process response: The program asserted that it has been systematically utilizing the results of assessment data for program improvement. However, the program noted that materials from the program's annual assessment retreats were not clearly identified during the visit. The program provided copies of the slides and minutes from their Spring 2011 Annual Program Assessment Retreat and copy of their 2009-2010 Program Assessment Report.

Due-process evaluation: The weakness has been resolved. The additional materials provided in the due-process response demonstrate that the program does annually review assessment data and, when appropriate, use that data for program improvement.

Program Observation

Wright State University's Management Information Systems program has developed and uses the web-based assessment management program called Assess My Program. This tool supports maintenance and tracking of assessment questions, assessment data, and program improvement decisions linked to assessment data. The tool has been presented at the annual ABET Symposium. The Department of Computer Science and Engineering may want to consider using this system for its own programs.

Engineering Accreditation Commission

Final Statement of Accreditation
to

Wright State University

Dayton, Ohio

2011-12 Accreditation Cycle

Leadership and Quality Assurance in Applied Science, Computing, Engineering, and Technology Education

ABET
ENGINEERING ACCREDITATION COMMISSION

WRIGHT STATE UNIVERSITY
Dayton, OH

FINAL STATEMENT
Visit Dates: November 6-8, 2011
Accreditation Cycle Criteria: 2011-12

Introduction & Discussion of Statement Construct

The Engineering Accreditation Commission (EAC) of ABET has evaluated the biomedical engineering, computer engineering, electrical engineering, engineering physics, industrial and systems engineering, materials science and engineering, and mechanical engineering programs of Wright State University.

This statement is the final summary of the EAC evaluation, at the institutional and engineering program levels. It includes information received during due process, including information submitted with the seven-day response. This statement consists of two parts: the first deals with the overall institution and its engineering operation, and the second deals with the individual engineering programs. It is constructed in a format that allows the reader to discern both the original visit findings and subsequent progress made during due process.

A program's accreditation action is based upon the findings summarized in this statement. Actions depend on the program's range of compliance or non-compliance with the criteria. This range can be construed from the following terminology:

- Deficiency: A deficiency indicates that a criterion, policy, or procedure is not satisfied. Therefore, the program is not in compliance with the criterion, policy, or procedure.

- Weakness: A weakness indicates that a program lacks the strength of compliance with a criterion, policy, or procedure to ensure that the quality of the program will not be

compromised. Therefore, remedial action is required to strengthen compliance with the criterion, policy, or procedure prior to the next evaluation.

- Concern: A concern indicates that a program currently satisfies a criterion, policy, or procedure; however, the potential exists for the situation to change such that the criterion, policy, or procedure may not be satisfied.

- Observation: An observation is a comment or suggestion that does not relate directly to the accreditation action but is offered to assist the institution in its continuing efforts to improve its programs.

Wright State University, founded in 1964 and granted full university status in 1967, is a regional university in the state of Ohio higher education system. Enrollment in the university is approximately 19,600 undergraduate students and over 3,000 graduate students. Undergraduate programs are offered in six colleges: Business, Education and Human Services, Engineering and Computer Science, Liberal Arts, Science and Mathematics, and Nursing and Health.

The College of Engineering and Computer Science (CECS) consists of four departments that offer the seven engineering programs under review: the Department of Biomedical, Industrial & Human Factors Engineering, the Department of Computer Science and Engineering, the Department of Electrical Engineering, and the Department of Mechanical and Materials Engineering. The college also offers eight master's programs and has 11 PhD focus areas in engineering and computer science. Enrollment in the college at the time of the visit was approximately 1,650 undergraduate and 400 graduate students. There are approximately 76 full-time faculty members in the college. The dean is new to the college and university, as of July 2010.

The following supporting units of the CECS were reviewed: Office of the Registrar, biology, chemistry, humanities/social sciences, library, mathematics, and physics. All supporting areas appear to adequately support the undergraduate engineering programs.

Institutional Strengths

1. There appears to be continued strong institutional support for the CECS. The dean, new to his office, appears to have the full support and confidence of the administration to carry out the mission of the college. The university has been approved as a center of excellence by the state in four areas including the engineering programs in the CECS.

2. The institution's physical plant, including the Russ Engineering Center and the attached Joshi Research Center is excellent. Buildings, classrooms, laboratories, faculty offices and public areas are well maintained and present a pleasing atmosphere conducive to the learning process.

3. The CECS enjoys a high level of industrial and government support for the engineering programs. The participation of the nearby industries in senior design projects and internships strengthens the engineering programs.

Computer Engineering
Program

Introduction

The computer engineering program is administered by the Department of Computer Science and Engineering that also offers a master's degree in computer engineering and bachelor's and master's degrees in computer science, and a doctoral degree in computer science and engineering. At the time of the visit the computer engineering program had 180 undergraduate students, 26 tenured or tenure-track faculty members and five part-time faculty members. The program graduated six students in the most recent academic year.

Program Strengths

1. The program has an outstanding faculty that shows commendable attention to students and to research. Faculty members benefit from close ties to the local industries and the nearby Air Force Research Lab.

2. The program has an excellent cooperative education program and incorporates input from the program as part of the assessment of student outcomes.

Program Weaknesses

1. Criterion 2. Program Educational Objectives This criterion requires that there be a documented and effective process involving program constituencies, for the periodic review and revision of the program educational objectives. The documented process to periodically review and revise the program educational objectives has not involved all of the program's constituencies. Without the involvement of all program constituencies, the program is unable to ensure its program educational objectives remain consistent with the needs of the program's constituents. Thus, the program lacks strength of compliance with this criterion.

    - Due–process response: The EAC acknowledges receipt of documentation related to the review and revision process for the program educational objects. The current review and revision process now provides for participation on the part of all the program's

constituencies. Students now participate via a student advisory board composed entirely of undergraduate computer science and computer engineering students. Since no input has been provided by the student advisory board for the program educational objectives, it is not clear how such information will be separated to provide reviews for both the computer science program and the computer engineering program. Therefore, it may not be possible to use the student advisory board input in a meaningful way for review of the computer engineering program educational objectives. Although a previous review process has been in place since 2006, the team was unable to view documents generated by the previous process during the on-site visit. In the due-process response the program indicated that the results of review of the program educational objectives in 2006, 2007, and 2011 resulted in changes to them. The due-process documentation did not contain reports or summary reports related to these prior reviews. Further, reports concerning prior reviews were not mentioned in the program's self-study report. Although there appears to be documentation of review of the program educational objectives that goes back to 2006, the program has not provided this data. Further, no data have been furnished from the student advisory board. Thus, the program lacks strength of compliance with this criterion.

- The weakness remains and will be a focus of the next review. In preparation for this review, the EAC anticipates evidence of a documented and effective process for the periodic review and revision of the program educational objectives

2. Criterion 4. Continuous Improvement This criterion requires that the program regularly use appropriate, documented processes for assessing and evaluating the extent to which both the program educational objectives and the student outcomes are being attained. The results of these evaluations must be systematically utilized as input for the continuous improvement of the program. Other available information may also be used to assist in the continuous improvement of the program. The program is gathering data for assessing and evaluating the extent to which the program educational objectives and the student outcomes are being attained. However, the results of the evaluations have not been systematically utilized as input for the continuous improvement of the program. Thus, the program lacks strength of compliance with this criterion.

- Due–process response: The EAC acknowledges receipt of documentation that demonstrates continuous improvements in the program resulting from the periodic assessment process of student outcomes.

- This weakness is resolved.

Program Concerns

1. Criterion 3. Student Outcomes This criterion requires the program to have documented student outcomes that prepare graduates to attain the program educational objectives. Student outcomes are outcomes (a) through (k) plus any additional outcomes that may be articulated by the program. Outcome (c) for the program did not list the realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability. Although student work indicated that some of these constraints were considered, the program did not include all of them in their outcome (c) list. Hence there is the potential that future compliance with this criterion could be jeopardized.

   - Due–process response: The EAC acknowledges receipt of documentation demonstrating that student outcomes (a) through (k) have been included in the program's student outcomes. The self-study report had inadvertently left out a phrase in the outcomes listing, but the assessment process considered all outcomes for this criterion.

   - The concern is resolved.

2. Criterion 5. Curriculum: This criterion requires that the students be prepared for engineering practice through a curriculum culminating in a major design experience based on the knowledge and skills acquired in earlier course work and incorporating appropriate engineering standards and multiple realistic constraints. Although the major design projects incorporate multiple realistic constraints, there is little evidence that engineering standards have been incorporated into the senior project. In particular, the course syllabus does not refer to the use of engineering standards. In an overall sense this criterion is satisfied. However, future compliance may be jeopardized if there is not adequate attention to appropriate engineering standards incorporated in the major design experience.

- Due–process response: The EAC acknowledges receipt of documentation demonstrating that changes have been made to ensure that students consider multiple realistic constraints including engineering standards. Standards are now listed in the course's syllabi and standards will be considered for each senior design project.

- The concern is resolved.

3. Program Criteria Program criteria for electrical, computer and similarly named engineering programs require that that the curriculum include applications of probability and statistics appropriate to the program name. A review of the course materials and discussion with the faculty provided limited evidence of such applications. Thus, there is the potential that future compliance with this criterion could be jeopardized.

   - Due–process response: The EAC acknowledges receipt of documentation demonstrating that the curriculum has three courses that include applications of probability and statistics appropriate to the program.

   - The concern is resolved.

Engineering Accreditation Commission

Final Statement of Accreditation
to

Wright State University

Dayton, OH

2013-2014 Accreditation Cycle

Assuring Quality • Stimulating Innovation

ABET
ENGINEERING ACCREDITATION COMMISSION

WRIGHT STATE UNIVERSITY
Dayton, OH

FINAL STATEMENT
Report submitted: July 01, 2013
Accreditation Cycle Criteria: 2013-2014

Introduction and Discussion of Statement Construct

The Engineering Accreditation Commission (EAC) of ABET has conducted an evaluation of the computer engineering program of Wright State University relative to shortcomings remaining after the 2011-12 general EAC review. The institution elected for this review to be conducted under the 2013-2014 Criteria for Accrediting Engineering Programs.

This statement is the final summary of the EAC evaluation. The first part of the statement addresses the institution and its overall engineering educational unit; the second addresses the computer engineering program. Its format allows the reader to discern both the original report findings and subsequent progress made during due process.

A program's accreditation action is based upon the findings summarized in this statement. Actions will depend on the program's range of compliance or non-compliance with the criteria. This range can be construed from the following terminology:

- Deficiency: A deficiency indicates that a criterion, policy, or procedure is not satisfied. Therefore, the program is not in compliance with the criterion, policy, or procedure.

- Weakness: A weakness indicates that a program lacks the strength of compliance with a criterion, policy, or procedure to ensure that the quality of the program will not be compromised. Therefore, remedial action is required to strengthen compliance with the criterion, policy, or procedure prior to the next review.

- Concern: A concern indicates that a program currently satisfies a criterion, policy, or procedure; however, the potential exists for the situation to change such that the criterion, policy, or procedure may not be satisfied.

- Observation: An observation is a comment or suggestion that does not relate directly to the current accreditation action but is offered to assist the institution in its continuing efforts to improve its programs.

Wright State University was founded in 1964 and granted full university status in 1967. It serves primarily as a regional university in the state of Ohio higher education system. Enrollment in the university is approximately 13,600 undergraduate students and over 3,400 graduate students. In addition to the College of Engineering and Computer Science (CECS), undergraduate programs are offered in Business, Education and Human Services, Liberal Arts, Science and Mathematics, and Nursing and Health. CECS consists of four departments that offer eight engineering programs: the Department of Biomedical, Industrial & Human Factors Engineering, the Department of Computer Science and Computer Engineering, the Department of Electrical Engineering, and the Department of Mechanical and Materials Engineering. The college offers 11 master's programs and has 8 Ph.D. focus areas in engineering and computer science. Enrollment in the college is currently 2,135 undergraduate and 834 graduate students. There are 78 full-time faculty members and 30 part-time faculty members in the college.

Computer Engineering
Program

Program Criteria for Electrical, Computer, and Similarly Named Engineering Programs

Introduction

The computer engineering program is administered by the Department of Computer Science and Engineering. The department also offers Bachelor of Science and Bachelor of Arts degrees in computer science, Master's degrees in computer engineering and computer science, and a Doctoral degree in computer science and engineering. Currently the computer engineering program has 248 undergraduate students. The department has 20 tenured or tenure-track faculty members and 12 part-time faculty members. In the most recent academic year, the department graduated 91 students in all three undergraduate degree programs, 26 of whom were in computer engineering. The institution elected for this review to be conducted under the 2013-2014 Criteria for Accrediting Engineering Programs.

Program Weakness

1. Criterion 2. Program Educational Objectives *The previous review cited that the program had published educational objectives that were not based on the input of all of the program constituencies. Specifically, the program's process of periodic review of these objectives did not offer substantive documentation that demonstrated the explicit inclusion of one of the identified constituencies, namely students currently enrolled in the bachelor of science computer engineering program. Criterion 2 requires that a program must have a documented, systematically utilized, and effective process, involving program constituencies, for the periodic review of these program educational objectives that ensures they remain consistent with the institutional mission, the program's constituents' needs, and the Engineering Criteria.*

   The report included a revised process flow chart for review of program educational objectives that explicitly identifies the role of input from computer engineering majors, as well as all other constituencies. Excerpts from the minutes of recent meetings of the student advisory board, which includes both computer engineering and computer science students, in March 2013, and the external advisory board, in November 2012 and June 2013, provided evidence that the program educational objectives have now been reviewed by all program constituencies.

Revision of the program educational objectives by the program based on documented input from all constituencies using the new process will be considered in fall 2013.

- The weakness is resolved.