CSCE 146: Algorithmic Design II

1. Course number and name: CSCE 146: Algorithmic Design II

2. Credit: 4-hrs; Contact: 2 lectures of 75 minutes each; 1 lab of 1 hr, 55 mins each per week

3. Instructor: Fall 2010: Jijun Tang, Spring 2011: Duncan Buell

4. Text book: Fall 2010--Simon Gray. *Data Structures in Java*, Pearson, 2007, ISBN: 9780321392794. Spring 2011—Duncan Buell. *Data Structures and Algorithms Using Java*, 2011, Jones and Bartlett, to appear (advance e-copy used).

5. Specific course information
   a. Catalog description: Continuation of CSCE 145. Rigorous development of algorithms and computer programs; elementary data structures.
      Note: Three lectures and two laboratory hours per week. Open to all majors.

   b. Prerequisites: Grade of C or better in both CSCE 145 and MATH 141

   c. Required in all curricula

6. Specific goals for the course
   a. Specific outcomes of instruction:
      1. Develop structured, modular algorithms.
      2. Implement correct programs in an object-oriented language.
      3. Use and implement as classes data structures, such as sets, bags, sequences, stacks, queues, and binary trees.
      4. Analyze the time and space complexity of simple algorithms. Introduce data abstraction and elementary concepts of object-oriented programming
      5. Implement moderately complex programs written in an object-oriented language.

   b. Relation of course outcomes to Student Outcomes: CE: see page 2; CS & CIS: see page 3

7. Topics covered and approximate weight (14 weeks, 4 hours/week, 56 hours total)
   1. Quick Overview of Object-Oriented Programming and Java
   2. Error Handling, Software Testing, and Program Efficiency
   3. Fundamental Data Structures: The Array and Linked Data Structures
   4. A Basic Collection Class
   5. The List Abstract Data Type
   6. The Stack Abstract Data Type
   7. The Queue Abstract Data Type
   8. Recursion
   9. Sorting and Searching
   10. Trees
   11. Binary Search Trees
   12. The Map ADT
   13. Graphs

Computer Engineering

Relation of Course Outcomes to EAC Student Outcomes*

	Student Outcomes											
Course Outcomes (CE)	(a) apply knowledge of mathematics, science, and engineering	(b) design and conduct experiments, … interpret data	(c) design a system, component, or process to meet desired needs …	(d) function on multidisciplinary teams	(e) identify, formulate, and solve engineering problems	(f) an understanding of professional and ethical responsibility	(g) communicate effectively	(h) the broad education to understand the impact of engineering solutions …	(i) a recognition of the need for, and an ability to engage in life-long learning	(j) a knowledge of contemporary issues	(k) use the techniques, skills, and modern engineering tools ….	(CE) demonstrate knowledge of discrete mathematics [CE]
Criteria	a	b	c	d	e	f	g	h	i	j	k	CE
1. Develop structured, modular algorithms.												
2. Implement correct programs in an object-oriented language												
3. Use and implement as classes data structures, such as sets, bags, sequences, stacks, queues, and binary trees.												
4. Analyze the time and space complexity of simple algorithms. Introduce data abstraction and elementary concepts of object-oriented programming												
5. Implement moderately complex programs written in an object-oriented language.												

* 3 = major contributor, 2 = moderate contributor, 1 = minor contributor; blank if not related

Computer Science & Computer Information Systems

Relation of Course Outcomes to CAC Student Outcomes*

	Student Outcomes											
	All									CS		CIS
Course Outcomes (CS & CIS)	(a) apply knowledge of computing and mathematics appropriate to the discipline	(b) analyze a problem, and identify and define the computing requirements …	c) design, implement, and evaluate a computer-based system, …	(d) function effectively on teams to accomplish a common goal	(e) An understanding of professional, ethical, legal, … responsibilities	(f) communicate effectively with a range of audiences	(g) analyze the local and global impact of computing on … society	(h) Recognition of the need for … continuing professional development	(i) current techniques, skills, and tools necessary for computing practice	(j) apply mathematical foundations, algorithmic principles, and CS theory …	(k) apply design and development principles	(j) An understand processes that support the information systems environment.
Criteria	a	b	c	d	e	f	g	h	i	j	k	j
1. Develop structured, modular algorithms.												
2. Implement correct programs in an object-oriented language												
3. Use and implement as classes data structures, such as sets, bags, sequences, stacks, queues, and binary trees.												
4. Analyze the time and space complexity of simple algorithms. Introduce data abstraction and elementary concepts of object-oriented programming												
5. Implement moderately complex programs written in an object-oriented language.												

* 3 = major contributor, 2 = moderate contributor, 1 = minor contributor; blank if not related