Bergen Community College
Computer Science Department
Course Syllabus

Instructor: ____________________________  Phone: __________________
Email: ________________________________  Office Hours ________________

Course Title: CIS-288 Discrete Mathematics for Computer Science

Prerequisites: CIS-265 or CIS-266

Credits/Hours: 4 Credits  4 Lecture

Gen’l Ed. Course: No

Course Description:
Discrete Mathematics for Computer Science is a study of the mathematical theory and techniques that underlie computer science. Topics considered include set theory; matrices; induction; counting techniques; relations; functions; recurrence relations; graphs and trees; Boolean algebra and circuits; grammars and languages; and an introduction to automata theory. Applications of these topics in computer science are included in the course.

Student Learning Outcomes: Upon completion of the course, the student will:

1. know the fundamental concepts of set theory, logic, induction, matrices, and combinatorics, and be able to apply them to the solution of problems related to computer science;
2. understand the fundamental concepts of relations, functions, and recurrence relations as they relate to applications in computer science;
3. be able to solve first and second order recurrence relations and use mathematical induction to verify solution;
4. know the fundamental properties and structure of graphs, networks, and trees, and be able to use this knowledge to solve problems that relate to computer science;
5. know the properties of Boolean algebras and be able to use them in the analysis and design of computer logic circuits;
6. understand the fundamental concepts of phrase structure grammars and programming languages.
7. be able to represent finite-state machines using state transition tables and state transition diagrams;
8. be able to apply the technique of machine minimization to improve the efficiency of a design

Student Learning Outcomes Assessment Measurement:
Each of the above listed student learning outcomes will be assessed by: (1) written assignments and/or quizzes; (2) written examinations and a comprehensive final exam.

Course grade: see the grading policy for the course.

Course Content:

1. Mathematical Preliminaries
   - set theory
   - computer representation of a finite set
   - counting techniques
   - matrix theory

2. Proof Techniques
   - propositions and theorems
   - disproof by counterexample
   - exhaustive proof
   - direct proof
   - proof by contraposition
   - proofs using the Principle of Mathematical Induction

3. Relations and Functions
   - binary relations: set builder, matrix, and digraph representation
   - properties of relations
   - equivalence relations and partitions
   - partial orders, Hasse Diagrams, and topological sort
   - operations on relations
   - functions: properties and operations
   - order of magnitude of functions

4. Recurrence Relations
   - examples of recurrence relations
   - classifying recurrence relations
   - solving first-order recurrence relations by iteration
   - verifying a solution using mathematical induction
   - solving second-order linear homogeneous recurrence relations
   - applications and computer implementation

5. Graphs, Networks, and Trees
   - fundamental terminology of graphs and trees
   - computer representation of graphs and trees
   - depth-first and breadth-first search
   - spanning trees and Prim’s algorithm
   - PERT / CPM techniques in systems analysis
   - decision trees and Huffman codes
   - Dijkstra’s shortest path algorithm

6. Boolean Algebras and Computer Logic
   - Boolean vectors, Boolean functions, and switching tables
   - Boolean operators and Boolean expressions
   - properties of a Boolean algebra
   - disjunctive normal form
   - minimization of Boolean expressions using properties
   - minimization of Boolean expressions using Karnaugh maps
   - applications: decision making constructs
   - applications: combinatorial circuits

7. Grammars and Languages
   - specification of a programming language
   - phrase structure grammars
• language generated by a grammar
• parsing and parse trees
• classification of grammars and languages
• Backus-Naur form
• syntax diagrams
• applications

8. Finite-State Machines
• components of a finite-state machine
• state transition functions, tables, and diagrams
• application: binary adder

9. Finite-State Automata
• finite state automata and recognition
• relationships between languages and finite automata
• machine minimization
• software simulation of a finite state automaton

References

Epp, Discrete Mathematics with Applications, PWS

Johnsonbaugh, Discrete Mathematics, Prentice-Hall

[ 8/2012 ]