DEPARTMENT OF COMPUTER SCIENCE
University of California, Davis

ECS 20 Discrete Mathematics for Computer Science 4 units

Format Lecture – 3 hours
Discussion/Laboratory – 1 hour

Catalog description

Discrete mathematics of particular utility to computer science. Proofs by induction. Propositional and first-order logic. Sets, functions, and relations. Big-O and related notations. Recursion and solutions of recurrence relations. Combinatorics. Probability on finite probability spaces. Graph theory.

Prerequisites Grade of C- or better in Mathematics 16A, 17A or 21A

Credit restrictions / cross listings None

Summary of course contents

1. Propositional and first-order logic
2. Elementary set theory.
3. Functions, relations, and equivalence relations.
4. Induction and recursion. Inductive definitions. Examples of induction taken from domains it is not obvious what one is inducting on.
5. The pigeonhole principle.
6. Asymptotic notation: O, Ω, Θ.
7. Derivation and solution of recurrence relations. Divide and conquer relations. Solutions through expansion and proofs by induction.
8. Combinatorics and counting. Permutations, combinations. Non-trivial counting problems.
9. Discrete probability
10. Graphs and trees. Terminology. Eulerian and Hamiltonian graphs. Graph colorings. Basic properties of trees.

Goals: Students will: (1) learn fundamental ideas and techniques from discrete mathematics; (2) improve their ability to write concise and rigorous proofs; (3) improve their ability to understand mathematical definitions and proofs; and (4) enhance their general mathematical sophistication.

Illustrative reading

- K. Rosen, *Discrete Mathematics and Its Applications*, 7th edition, McGraw-Hill, 2011.
- M. Lipson, *Schaum's Outline of Discrete Mathematics*, revised 3rd edition, 2009.
- D. Velleman, *How to Prove it: A Structured Approach*. Cambridge University Press, 1994.

GE3 Science & Engineering

Quantitative Literacy

Overlap The course overlaps with Math 108 (Introduction to Abstract Mathematics) but focusses more on problems and techniques of utility in computer science. The course includes topics like asymptotic notation, graphs, and trees, which Math 108 does not include, while it omits topics like cardinal numbers, which Math 108 does include. ECS 20 is a fundamental, preparatory course for upper-division ECS courses.

Instructors Staff

History 2012.10.11 (P. Rogaway): Replaced catalog description and summary of course contents with more accurate versions; revised overlap description. Prior course description went back to May 2001 (Z. Bai, D. Gusfield, P. Rogaway).

DEPARTMENT OF COMPUTER SCIENCE
University of California, Davis

Outcomes

1	✓	an ability to apply knowledge of mathematics, science, computing, and engineering
2		an ability to design and conduct experiments, as well as to analyze and interpret data
3		an ability to design, implement, and evaluate a system, process, component, or program to meet desired needs, within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability
4		an ability to function on multi-disciplinary teams
5		an ability to identify, formulate, and solve computer science and engineering problems and define the computing requirements appropriate to their solutions
6		an understanding of professional, ethical, legal, security and social issues and responsibilities
7		an ability to communicate effectively with a range of audiences
8		the broad education necessary to understand the impact of computer science and engineering solutions in a global and societal context
9	✓	a recognition of the need for, and an ability to engage in life-long learning
10		knowledge of contemporary issues
11	✓	an ability to use current techniques, skills, and tools necessary for computing and engineering practice
12		an ability to apply mathematical foundations, algorithmic principles, and computer science and engineering theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
13		an ability to apply design and development principles in the construction of software systems or computer systems of varying complexity