CS G262: DISCRETE STRUCTURES

Item	Value
Curriculum Committee Approval Date	05/02/2023
Top Code	070600 - Computer Science (Transfer)
Units	3 Total Units
Hours	54 Total Hours (Lecture Hours 54)
Total Outside of Class Hours	0
Course Credit Status	Credit: Degree Applicable (D)
Material Fee	No
Basic Skills	Not Basic Skills (N)
Repeatable	No
Grading Policy	Standard Letter (S)
California General Education Transfer Curriculum (Cal-GETC)	• Cal-GETC 2A Math Concepts (2A)
Intersegmental General Education Transfer Curriculum (IGETC)	• IGETC 2A Math Concepts (2A)
California State University General Education Breadth (CSU GE-Breadth)	• CSU B4 Math/Quant.Reasoning (B4)

Course Description

This course is an introduction to the discrete structures used in Computer Science with an emphasis on their applications. Topics covered include: Functions, Relations and Sets; Basic Logic; Proof Techniques; Basics of Counting; Graphs and Trees; and Discrete Probability. PREREQUISITE: CS G153 or CS G175; and course taught at the level of intermediate algebra or appropriate math placement. Transfer Credit: CSU; UC. C-ID: COMP 152. **C-ID:** COMP 152.

Course Level Student Learning Outcome(s)

1. ILOs
2. iSLO 1. Specialized Subject Knowledge (Majors) - Demonstrate a depth of knowledge, skills, and abilities in a particular major.
3. iSLO 3. Analytic skills - Identify, evaluate, and apply a variety of methods to solve problems.
4. iSLO 4. Information competency skills - Determine the scope of information needs; locate and retrieve relevant information; organize, analyze, and evaluate information; and understand the ethical and legal issues surrounding information and information technology.
5. iSLO 5. Quantitative skills - Convert information into relevant symbolic and mathematical forms (e.g. equations, graphs, diagrams, tables), provide accurate explanations of information presented in mathematical forms, and successfully perform calculations and symbolic operations.
6. Course Outcomes
7. Describe how formal tools of symbolic logic are used to model real-life situations, including those arising in computing contexts such as program correctness, database queries, and algorithms.
8. Relate the ideas of mathematical induction to recursion and recursively defined structures.
9. Analyze a problem to create relevant recurrence equations.
10. Demonstrate different traversal methods for trees and graphs.
11. Apply the binomial theorem to independent events and Bayes' theorem to dependent events.

Course Objectives

- 1. Construct truth tables using propositional logic and logical connectives.
- 2. Explain sets, functions, and sequences and summations and their applications.
- 3. Solve counting problems using the Pigeonhole Principle, permutations, and combinations.
- 4. Calculate discrete probability problems including conditional probability, Bayes Theorem, and mathematical expectation.
- 5. Construct recursive definitions and recursive algorithms.
- 6. Define graph terminology and types of graphs and use this terminology to solve graph theory problems including the existence of Euler and Hamilton circuits and paths, shortest-path problems, and graph coloring.
- 7. Define tree terminology and identify applications of trees.
- 8. Perform tree traversal techniques and find spanning trees and minimum spanning trees.
- 9. Define regular sets and show how languages can be recognized by finite-state automata and Turing machines.
- 10. Solve counting problems using Inclusion-Exclusion.
- 11. Solve recurrence relations.

Lecture Content

Functions, Relations and Sets Functions (surjections, injections, inverses, composition) Relations (reflexivity, symmetry, transitivity, equivalence relations) Sets (Venn diagrams, complements, Cartesian products, power sets) Pigeonhole principles Cardinality and countability Basic Logic Propositional logic Logical connectives Truth tables Normal forms (conjunctive and disjunctive) Validity Predicate logic Universal and existential quantification Modus ponens and modus tollens Limitations of predicate logic Proof Techniques Notions of implication, converse, inverse, contrapositive, negation, and contradiction The structure of mathematical proofs Direct proofs Proof by counterexample Proof by contradiction Mathematical induction Strong induction Recursive mathematical definitions Well orderings Basics of Counting Counting arguments Sum and product rule Inclusion-exclusion principle Arithmetic and geometric progressions Fibonacci numbers The pigeonhole principle Permutations and combinations Basic definitions Pascals identity The binomial theorem Solving recurrence relations Common examples The Master theorem Graphs and Trees Trees Undirected graphs Directed graphs Spanning trees/forests Traversal strategies Discrete Probability Finite probability space, probability measure, events Conditional probability, independence, Bayes theorem Integer random variables, expectation Law of large numbers

Lab Content

No lab required.

Method(s) of Instruction

- Lecture (02)
- DE Live Online Lecture (02S)
- DE Online Lecture (02X)

Reading Assignments

Textbook and instructor provided handouts.

Writing Assignments

Create diagrams and reports on procedures for computer programs and theory applications.

Out-of-class Assignments

Students will create programming and theoretical solutions for problems assigned in class.

Demonstration of Critical Thinking

Students will analyze computational problems and provide algorithmic solutions that will have the proper mathematical support for correctness and efficiency like finding a loop invariant or defining the growth function for an algorithm complexity.

Required Writing, Problem Solving, Skills Demonstration

Students will be required to solve computational problems through algorithms that are correct, finite, and practical.

Eligible Disciplines

Computer science: Masters degree in computer science or computer engineering OR bachelors degree in either of the above AND masters degree in mathematics, cybernetics, business administration, accounting or engineering OR bachelors degree in engineering AND masters degree in cybernetics, engineering mathematics, or business administration OR bachelors degree in mathematics AND masters degree in cybernetics, engineering mathematics, or business administration OR bachelors degree in any of the above AND a masters degree in information science, computer information systems, or information systems OR the equivalent. Note: Courses in the use of computer programs for application to a particular discipline may be classified, for the minimum qualification purposes, under the discipline of the application. Masters degree required.

Textbooks Resources

1. Required Jenkyns, T., Stephenson, B. Fundamentals of Discrete Math for Computer Science: A Problem-Solving Primer (Undergraduate Topics in Computer Science), 2nd (latest) ed. Springer, 2018 Rationale: New textbook edition.