COURSE OUTLINE OF RECORD

Number: CS G262  TITLE: Discrete Structures

ORIGINATOR: Cristian Racataian  EFF TERM: Fall 2014
FORMERLY KNOWN AS:
DATE OF
OUTLINE/REVIEW: 05-04-2017
CROSS LISTED COURSE:
TOP NO: 0706.00
CID: COMP 152

SEMESTER UNITS: 3.0
HRS LEC: 54.0  HRS LAB: 0.0  HRS OTHER: 0.0
CONTACT HRS TOTAL: 54.0
STUDY NON-CONTACT HRS RECOMMENDED: 108.0

CATALOG 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.

JUSTIFICATION FOR COURSE:

PREREQUISITES:
- CS G153: Java Programming, Introduction with a minimum grade of C or better
  or
- CS G175: C++ Programming with a minimum grade of C or better
  and
- MATH G030: Intermediate Algebra with a minimum grade of C or better or Mathematics Placement Assessment
  or
- MATH G040: Accelerated Elementary and Intermediate Algebra with a minimum grade of C or better

COREQUISITES:

ADVISORIES:

ASSIGNED DISCIPLINES:
- Computer science

MATERIAL FEE: Yes [ ] No [X] Amount: $0.00

CREDIT STATUS: Noncredit [ ] Credit - Degree Applicable [X] Credit - Not Degree Applicable [ ]

GRADING POLICY: Pass/No Pass [ ] Standard Letter [X] Not Graded [ ] Satisfactory Progress [ ]

OPEN ENTRY/OPEN EXIT: Yes [ ] No [X]

TRANSFER STATUS: CSU Transferable[ ] UC/CSU Transferable[X] Not Transferable[ ]

BASIC SKILLS STATUS: Yes [ ] No [X] LEVELS BELOW TRANSFER: Not Applicable

CALIFORNIA CLASSIFICATION CODES: Y - Not Applicable

NON CREDIT COURSE CATEGORY: Y - Not applicable, Credit Course

OCCUPATIONAL (SAM) CODE: E

REPEATABLE ACCORDING TO STATE GUIDELINES: No [X] Yes [ ] NUMBER REPEATS:

REQUIRED FOR DEGREE OR CERTIFICATE: No [ ] Yes [X]

Computer Science(Associate in Science for Transfer)

GE AND TRANSFER REQUIREMENTS MET:
PROGRAM LEVEL LEARNING OUTCOME(S) Supported by this course:

Describe formal tools of symbolic logic as they relate to real-life situations, program correctness, database queries, and algorithms.

Design and implement algorithms that include basic computation techniques, simple I/O, conditional and iterative structures, and the definition of functions.

COURSE LEVEL STUDENT LEARNING OUTCOME(S) Supported by this course:

1. 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.
2. Relate the ideas of mathematical induction to recursion and recursively defined structures.
3. Analyze a problem to create relevant recurrence equations.
4. Demonstrate different traversal methods for trees and graphs.
5. 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.
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.

COURSE CONTENT:
LECTURE CONTENT:

A. Functions, Relations and Sets
   1. Functions (surjections, injections, inverses, composition)
   2. Relations (reflexivity, symmetry, transitivity, equivalence relations)
   3. Sets (Venn diagrams, complements, Cartesian products, power sets)
   4. Pigeonhole principles
   5. Cardinality and countability

B. Basic Logic
   1. Propositional logic
   2. Logical connectives
   3. Truth tables
   4. Normal forms (conjunctive and disjunctive)
   5. Validity
   6. Predicate logic
   7. Universal and existential quantification
   8. Modus ponens and modus tollens
   9. Limitations of predicate logic

C. Proof Techniques
   1. Notions of implication, converse, inverse, contrapositive, negation, and contradiction
   2. The structure of mathematical proofs
   3. Direct proofs
   4. Proof by counterexample
   5. Proof by contradiction
   6. Mathematical induction
   7. Strong induction
   8. Recursive mathematical definitions
   9. Well orderings

D. Basics of Counting
   1. Counting arguments
   2. Sum and product rule
   3. Inclusion-exclusion principle
   4. Arithmetic and geometric progressions
   5. Fibonacci numbers
   6. The pigeonhole principle
   7. Permutations and combinations
   8. Basic definitions
   9. Pascal’s identity
   10. The binomial theorem
   11. Solving recurrence relations
   12. Common examples
   13. The Master theorem

E. Graphs and Trees
   1. Trees
   2. Undirected graphs
   3. Directed graphs
   4. Spanning trees/forests
   5. Traversal strategies

F. Discrete Probability
   1. Finite probability space, probability measure, events
   2. Conditional probability, independence, Bayes’ theorem
   3. Integer random variables, expectation
   4. Law of large numbers
METHODS OF INSTRUCTION:
A. Lecture:
B. Dist. Ed – Delayed Interaction:
C. Online:

INSTRUCTIONAL TECHNIQUES:
Lecture and/or discussion for specific content topics using sample code, projector and white board. Hands-on programming projects and theory application during lecture.

COURSE ASSIGNMENTS:
Out-of-class Assignments
Students will create programming and theoretical solutions for problems assigned in class.

Writing Assignments
Diagrams and reports on procedures for computer programs and theory applications.

Reading Assignments
Text book and instructor provided handouts.

METHODS OF STUDENT EVALUATION:
Midterm Exam
Final Exam
Short Quizzes
Written Assignments
Objective Examinations
Projects (ind/group)
Problem Solving Exercises
Oral Presentations
Skills Demonstration

Demonstration of Critical Thinking:
Students will be demonstrating their projects. Optional research papers and classroom presentations will further demonstrate their ability in critical thinking and problem solving.

Required Writing, Problem Solving, Skills Demonstration:
Students will be required to complete programming projects presented to them in the form of business automation problems requiring solution implementation. Students will be required to write documentation for their projects.

TEXTS, READINGS, AND RESOURCES:
TextBooks:

LIBRARY:
Adequate library resources include: Print Materials

Comments:

Attachments:
Attached Files