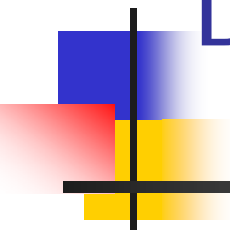


We Claim this Class for Computer Science: A Non-Mathematician's Discrete Structures Course



Adrienne Decker

Department of Computer
Science & Engineering
University at Buffalo, SUNY
adrienne@cse.buffalo.edu

Phil Ventura

Department of Computer
Science
State University of West
Georgia
pventura@westga.edu



Institution Background

- One-semester Discrete Mathematics course required by all majors
- Course has 4 hours of contact time per week: 3 hours lecture, 1 hour recitation
- Departmental requirement of a one-semester probability/statistics course



Conflict Arises

- Current version of Discrete Math not working
- Unhappy students
- Unhappy faculty in upper-level courses



Proposed Solution

- CC2001 Guidelines
- Make the course speak to the students



Reorganization

- Topics covered in Discrete Structures
 - Logic, including formal logic proofs
 - Functions, Relations, Sets
 - Proof Techniques
 - Counting (Recurrence Relations Only)
 - Graphs and Trees



Topic Presentation

- General introduction to topic
 - Terms
 - Definitions
- Mathematical Formalism Developed
 - Theorems/Axioms
 - Proofs where appropriate
 - Homework Problems Assigned
- Applications



Logic

- Basic Inference and Equivalence Rules
- Propositional Logic Proofs
- Predicate Logic
- Translations
- Predicate Logic Proofs

- Application
 - Prolog



Sets, Relations, Functions

- ADTs
- Created Set.java

Set

java.util.HashSet _set

Set()

void insert(Object obj)

boolean member(Object obj)

boolean equals(Set s)

boolean isEmptySet()

boolean isSubset(Set sub)

Set union(Set other)

Set intersection(Set other)

Set difference(Set other)

Set cartesianProduct(Set other)

Set powerSet()

int cardinality()

String toString()



Sets, Relations, Functions

- Created Relation.java

Relation

Relation()

boolean isOneToOne()

boolean isOneToMany()

boolean isManyToOne()

boolean isManyToMany()

boolean isReflexive()

boolean isSymmetric()

boolean isAntiSymmetric()

boolean isTransitive()

boolean isPartialOrdering()

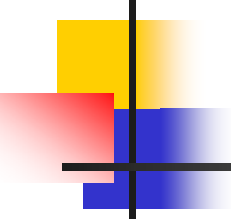
boolean isEquivalenceRelation()

void reflexiveClosure()



Sets, Relations, Functions

- Functions Introduced
 - Future (this semester?): Introduce Functional Programming Language



Recurrence Relations, Induction, and Recursion – oh my!

- Topics are related
- Many students familiar with recursion
- Induction Proofs
 - Not just natural numbers!
 - Structural induction over Strings, Trees, and Graphs



Graphs and Trees

- Trees discussed extensively in CS2
 - Decision Trees
 - Huffman Codes
- Graphs focused on more heavily
 - Warshall's Algorithm
 - Euler Path, Hamiltonian Circuit
 - Shortest Path
 - Minimum Spanning Tree
 - Depth-First Search, Breadth-First Search



Observations

- Students responded to applications
- Students commented that material overlapped
- Wanted theoretical treatment of concepts covered in other courses



Experiments

- Hypothesis: Students taking both Discrete Structures and CS2 would perform better in CS2 than those who do not.
- Hypothesis: Students taking both Discrete Structures and CS2 would perform better in Discrete Structures than those who do not.



Results

- No benefit to student grades in CS2
- Benefits for student grades in Discrete Structures



Open Issues

- Introduction of a functional programming language
- Graph Visualization
- Re-ordering of topics of course
- CS1 prerequisite