A: *Division:* **Instructional**

B: Faculty: **Pure and Applied Science and Technology**

Date: **May , 1997**

New Course:

Revision of Course: **X**

Dated: **June 11, 1996**

C: **MATH 130** *Course Number*

D: **Discrete Mathematics I** *Descriptive Title*

E: **3** *Credits*

F: *Calendar Description:*

This is the first of two Discrete Mathematics courses for Computing Science students. Topics include logic, set theory, counting, functions, relations, graphs, trees, finite state machines, formal languages and Boolean algebra.

Summary of Revisions

Sections **N, O, P** (more detail)

G: *Type of Instruction: Hours per Week*

Lecture	4
Laboratory	
Seminar	
Clinical Experience	
Field Experience	
Practicum	
Shop	
Studio	
Student Directed Learning	
Other	
TOTAL	4 HOURS

H: *Course Prerequisites:*

Math 12 (C or better)

I: *Course Corequisites:*

None

J: *Course for which this course is a prerequisite:*

MATH 230

K: *Maximum Class Size:*

35

L: *College Credit*

Transfer **X**

Non-Transfer

M: *Transfer Credit:*

Requested

Granted **X**

Course Equivalents:

U.B.C.

S.F.U. **MACM 101**

U.Vic

Course Designer

Vice President

Dean

Registrar

N: *Textbook and Materials to be Purchased by Students:*

Rosen, H.R., Discrete Mathematics and its Applications, McGraw Hill, 1995.

O: *Course Objectives:*

The student should be able to:

- write english statements in symbolic form using propositional variables or functions, logical connectives and any necessary quantifiers;
- determine the truth value of a statement under an interpretation;
- determine the negation, converse or contrapositive of a statement;
- verify logical equivalencies;
- demonstrate an understanding of tautologies, contradictions and duals;
- prove the properties of logic;
- determine the cardinality of sets, subsets, power sets and Cartesian products;
- combine sets using the set operators;
- prove set identities by showing that each expression is a subset of the other;
- use membership tables or Venn diagrams to prove set identities;
- classify functions as injective, surjective or bijective;
- demonstrate an understanding of domains, codomains, ranges, mappings and images;
- create new functions by composition;
- find the inverse of an injective function;
- demonstrate and understanding of the floor and ceiling functions;
- compute finite sums;
- determine if a set is countable or uncountable;
- give a big-O estimate for a function;
- write a simple algorithm in pseudocode;
- determine the time complexity of simple algorithms;
- demonstrate an understanding of divisibility, the greatest common divisor and modular arithmetic;
- use the Euclidean algorithm to find the gcd of two numbers;
- convert between binary, octal and hexadecimal;
- find the sum, difference, product, join, meet, and Boolean product of two matrices;
- demonstrate an understanding of the rules of inference;
- analyze an argument as to its validity using the concepts of mathematical logic;
- use a direct proof, indirect proof, or contradiction to prove a mathematical theorem;

- prove mathematical theorems using formal inductive techniques;
- give a recursive definition of a function or set;
- use the sum and product rule and tree diagrams to solve basic counting problems;
- apply the inclusion-exclusion principle to solve counting problems for two tasks.
- solve counting problems using the Pigeon-Hole Principle;
- count unordered selections of distinct objects;
- count ordered arrangements of objects of a finite set;
- find the expansion of a binomial;
- determine the probability of a combination of events for an equi-probable sample space;
- determine whether a relation is reflexive, irreflexive, symmetric, antisymmetric, asymmetric, and/or transitive;
- combine relations and form the composite of two or more relations;
- find the inverse and complement of a relation;
- determine the projection and join of two n-ary relations;
- represent a relation as a matrix and a digraph;
- find the reflexive, symmetric, and transitive closures of a relation;
- identify the various types of graphs;
- draw graph models;
- demonstrate an understanding of the vocabulary of graph theory;
- determine whether a graph is bi-partite;
- represent a graph as an adjacency matrix and an incidence matrix;
- determine whether a pair of graphs are isomorphic;
- find circuits and paths in a graph;
- distinguish between a tree and a graph;
- describe the components and properties of various types of trees;
- prove or disprove Boolean identities;
- manipulate Boolean expressions;
- find the dual of a Boolean expression;
- find the sum-of-products expansion of a Boolean function;
- determine whether a string belongs to the language generated by a given grammar;
- classify a grammar;
- find the language created by a grammar;
- draw the state diagram for a finite-state machine;
- construct a finite-state machine to perform a function;
- determine the output of a finite state machine;

P: *Course Content:*

1. **Logic**
   - 1.1. Propositions and truth tables.
   - 1.2. Logical equivalencies.
   - 1.3. Predicates and quantifiers.
   - 1.4. The laws of logic.

2. **Set Theory**
   - 2.1. Cardinality.
   - 2.2. Combining sets.
   - 2.3. Venn diagrams and membership tables.
   - 2.4. The laws of set theory.
   - 2.5. Countable and uncountable sets.

3. **Functions**
   - 3.1. Injective, surjective and bijective functions.
   - 3.2. Composites and inverses.
   - 3.3. The floor and ceiling functions.
   - 3.4. Sequences and sums.
   - 3.5. The growth of functions and Big-O notation.

4. **Algorithms, Integers and Matrices**
   - 4.1. Introduction to algorithms and their complexity.
   - 4.2. Introduction to number theory.
   - 4.3. Matrices.

5. **Mathematical Reasoning and Recursive Definitions**
   - 5.1. Rules of inference
   - 5.2. Deductive and Inductive proofs
   - 5.3. Recursive Definitions

6. **Counting**
   - 6.1. Fundamental counting principles.
   - 6.2. The pigeonhole principle.
   - 6.3. Permutations and combinations.
   - 6.4. Binomial theorem.
   - 6.5. Introduction to probability.

7. **Relations**
   - 7.1. Properties of relations.
   - 7.2. Combining relations.
   - 7.3. Digraph and matrix representations.
   - 7.4. Closures of relations.

8. **Graphs and Trees**
   - 8.1. Definitions, terminology and properties.
   - 8.2. Matrix representations.
   - 8.3. Isomorphism.

9. **Boolean Algebra**
   - 9.1. Boolean expressions and functions.
   - 9.2. Representing Boolean functions.
   - 9.3. Functional completeness.

10. **Languages and Finite State Machines**
   - 10.1. Languages and grammars.
   - 10.2. Finite-state machines with output.
   - 10.3. Finite-state automata.

Q: *Method of Instruction*

Lectures, problem sessions and assignments.

R: *Course Evaluation:*

Evaluation will be carried out in accordance with Douglas College policy. The instructor will present a written course outline with specific evaluation criteria at the beginning of the semester. Evaluation will be based on some of the following:

Weekly tests	{ 0 - 40% }
Midterm tests	{ 20 - 70% }
Assignments	{ 0 - 15% }
Attendance	{ 0 - 5% }
Class participation	{ 0 - 5% }
Final Examination	{ 30% }