



A: *Division:* Instructional *Date:* May, 1997  
 B: *Faculty:* Science and Technology *New Course:* X  
*Revision of Course:* \_\_\_\_\_  
*Dated:* \_\_\_\_\_

C: MATH 230 D: Discrete Mathematics II E: 3  
*Course Number* *Descriptive Title* *Credits*

F: *Calendar Description:*

This is the second of two Discrete Mathematics courses for Computing Science students. Topics include complexity of algorithms, recursion, recurrence relations, generating functions, equivalence relations, partial orders, partitions, graphs and trees, cycles and paths, shortest-path algorithms, minimal spanning trees, tree traversals and applications of trees and graphs.

*Summary of Revisions*

G: *Type of Instruction: Hours per Week*

Lecture	<u>4</u>
Laboratory	_____
Seminar	_____
Clinical Experience	_____
Field Experience	_____
Practicum	_____
Shop	_____
Studio	_____
Student Directed Learning	_____
Other	_____
<b>TOTAL</b>	<b><u>4</u> HOURS</b>

H: *Course Prerequisites:*  
**Math 130**

I: *Course Corequisites:*  
**None**

J: *Course for which this course is a prerequisite:*  
**None**

K: *Maximum Class Size:*  
**35**

L: *College Credit*

Transfer X  
 Non-Transfer \_\_\_\_\_

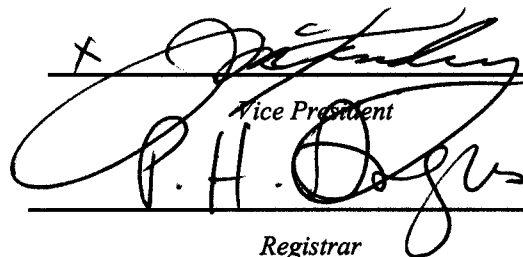
M: *Transfer Credit:*

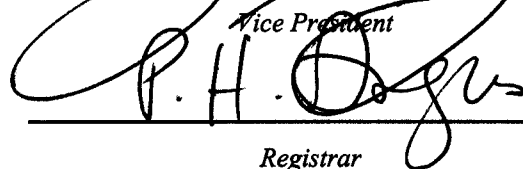
Requested X  
 Granted \_\_\_\_\_

*Course Equivalents:*

U.B.C.  
 S.F.U. **MACM 201**  
 U.Vic

  
 \_\_\_\_\_  
 Course Designer

 *Sept 5, 97.*  
 \_\_\_\_\_  
 Vice President

  
 \_\_\_\_\_  
 Registrar

  
 \_\_\_\_\_  
 Dean



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:

- determine whether a set is countable or uncountable;
- demonstrate Cantor's diagonalization process;
- devise recursive algorithms and compare with iterative algorithms;
- use a loop invariant to prove that a program segment is correct;
- determine the worst case and average case complexity of a simple algorithm;
- determine the number of ordered and unordered selections of  $r$  elements chosen with and without repetition from a set with  $n$  elements;
- determine the permutations of sets with indistinguishable objects;
- enumerate the ways distinguishable objects can be placed into distinguishable boxes;
- develop an algorithm to generate permutations of a set;
- develop a recurrence relation to model a problem;
- solve recurrence relations iteratively;
- solve linear homogeneous recurrence relations with constant coefficients of degree two;
- verify solutions to linear inhomogeneous recurrence relations;
- determine the big-O of divide-and-conquer recurrence algorithms such as the binary search;
- apply the inclusion-exclusion principle to problems with more than 2 sets;
- use the principle of inclusion-exclusion to solve counting problems modeled after the problem of finding the number of integer solutions of a linear equation with constraints;
- solve counting problems modeled after the number of onto functions from one finite set to another;
- count the number of derangements of a set and solve counting problems based on this principle;
- derive generating functions for a sequence;
- use ordinary and exponential generating functions to solve counting problems;
- use a generating function to solve a recurrence relation;
- determine if a relation is an equivalence relation;
- determine the equivalence classes of an equivalence relation;



- determine if a collection of subsets is a partition of a given set;
- demonstrate an understanding of partial order, total order, lexicographic order, well-ordered, maximal element, minimal element, greatest element, least element, bounds, lattice;
- draw a Hasse diagram of a poset;
- represent a digraph as an adjacency matrix or incidence matrix;
- determine whether a pair of graphs are isomorphic;
- demonstrate an understanding of connected, simple path, weighted graph, circuit, subgraph, cut vertices, cut edges, degree of a vertex;
- determine whether a graph has an Euler path or circuit;
- determine whether a graph has an Hamiltonian path or cycle;
- model and solve problems with a graph;
- use Dijkstra's algorithm to find the shortest path between pairs of vertices in a weighted graph;
- demonstrate an understanding of the terminology of rooted trees and of  $m$ -ary trees;
- form a binary search tree using a recursive algorithm and analyze the algorithm;
- model a problem using a decision tree;
- construct a binary tree which represents a given prefix coding scheme;
- find the word represented by a bit string given a coding scheme;
- demonstrate an understanding of a Huffman tree;
- demonstrate an understanding of three algorithms for traversing all vertices of an ordered rooted tree (preorder traversal, inorder traversal, postorder traversal);
- represent a complicated expression using a binary tree and write the expression in prefix, postfix, or infix notation;
- evaluate a prefix, postfix, or infix expression;
- demonstrate an understanding of several sorting algorithms and their complexity (bubble sort, merge sort, selection sort, quick sort);
- find all the nonisomorphic spanning trees of a graph;
- use a depth-first or breadth-first search to produce a spanning tree;
- use Prim's algorithm or Kruskal's algorithm to construct a minimum spanning tree for a weighted graph.



P: *Course Content:*

1. **Infinite Sets, Computability and Recursion**

- 1.1. Cardinality of infinite sets.
- 1.2. Recursion and iteration.
- 1.3. Complexity of algorithms.
- 1.4. Program correctness.

2. **Advanced Counting**

- 2.1. Permutations and combinations with repetition.
- 2.2. Indistinguishable and distinguishable objects.
- 2.3. Recurrence relations.
- 2.4. Solving first and second order linear recurrence relations.
- 2.5. Generating functions.
- 2.6. Solving recurrence relations using generating functions.
- 2.7. Solving counting problems using generating functions.
- 2.8. Divide-and-conquer relations.
- 2.9. Applications of inclusion-exclusion.

3. **Relations**

- 3.1. Equivalence relations and partitions.
- 3.2. Partial orderings and Hasse diagrams.

4. **Graphs**

- 4.1. Representations.
- 4.2. Connectivity.
- 4.3. Euler and Hamilton paths.
- 4.4. Shortest path problems.

5. **Trees**

- 5.1. Applications.
- 5.2. Tree traversals.
- 5.3. Trees and sorting.
- 5.4. Spanning trees.
- 5.5. Minimum spanning trees.



*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% }