Foundations Course Articulation Renewal Proposal
ICS-141: *Discrete Mathematics for Computer Science I*
As a Symbolic Reasoning (FS) Course

Leeward Community College
Fall 2007

Page	Contents

I. Course Description, Student Learning Outcomes.

- Title & Catalog Description

ICS-141: Discrete Mathematics for Computer Science I (3 credits): Includes logic, sets, functions, matrices, algorithmic concepts, mathematical reasoning, recursion, counting techniques, probability theory. * (45 lecture hours)

- Student Learning Outcomes

Upon completion of ICS-141, the student should be able to

Analyze issues and apply mathematical problem solving skills to plan courses of actions in decision-making situations, using:

1. Basic mathematical formal logic.
2. Proofs.
3. Recursion.
4. Analysis of algorithms.
5. Sets.
6. Combinatorics.
7. Relations.
8. Functions.
9. Matrices.
10. Probability.

II. Changes.

No significant changes have been made in ICS-141 since the original request for foundations designation was approved. The Student Learning Outcomes were re-worded for clarity, but the content of the course remains the same.

III. Assessing the Course: Below are samples of course materials that illustrate how the course meets the Symbolic Reasoning (FS) Foundations Hallmark.

1. *Students will be exposed to the beauty, power, clarity and precision of formal systems. How will the course meet this hallmark?*

The breadth of material covered in ICS-141 includes many of the major topics of mathematics and computer science theory. They include logic, sets, functions, matrices, mathematical reasoning and counting techniques. Faculty in the ICS discipline will introduce applications of these topics to everyday problems. These application examples are carefully selected to expose the beauty, power, clarity and precision of formal computational systems. In addition, the homework exercises, and quiz and exam problems are specifically selected to show students the power and precision of these computational systems.

For example, in the coverage of logic, students are required to learn a basic set of symbols representing Boolean operations that can be performed to build propositions. The following table summarizes some of the symbols used:

Formal Name	**Nickname**	**Arity**	**Symbol**
Negation operator	NOT	Unary	$\neg$
Conjunction operator	AND	Binary	$\wedge$
Disjunction operator	OR	Binary	$\vee$
Exclusive-OR operator	XOR	Binary	$\oplus$
Implication operator	IMPLIES	Binary	$\rightarrow$
Biconditional operator	IFF	Binary	$\leftrightarrow$

Using these symbols and Boolean operations as building blocks, they can solve problems such as:

1. Complete a truth table for: $\neg p \vee (p \wedge q)$.

2a. Use a truth table to determine if: $\neg(p \leftrightarrow q) \equiv \neg p \leftrightarrow q$

2b. Is $\neg(p \leftrightarrow q)$ a tautology, contradiction or a contingency?

3. Use logical equivalence rules to show: $(p \wedge \neg q) \vee (p \wedge q) \equiv p$

The students are taught to produce two solutions to problems such as the ones that follow on the next page.

1.

p	q	$\neg p$	$p \wedge q$	$\neg p \vee (p \wedge q)$
T	T	F	T	T
T	F	F	F	F
F	T	T	F	T
F	F	T	F	T

2a.

p	q	$\neg p$	$p \leftrightarrow q$	$\neg(p \leftrightarrow q)$	$\neg p \leftrightarrow q$
T	T	F	T	F	F
T	F	F	F	T	T
F	T	T	F	T	T
F	F	T	T	F	F

The propositions $\neg(p \leftrightarrow q)$ and $\neg p \leftrightarrow q$ are logically equivalent because their truth values are the same on all four lines of the truth table. In programming courses, students are expected to be able to develop complex Boolean expressions to implement solutions to algorithms. The study of logic helps students develop the skills they will need when writing program code. Logical equivalences are shown to the student so they can see that it is often the case that Boolean expressions can be created differently but still yield the correct solution to the problem. The design of digital logic circuits is another example where logical equivalences are used. Students learn to design separate circuits that produce the same output given a certain input stream because they are logically equivalent.

2b. The proposition $\neg(p \leftrightarrow q)$ is a contingency since it has a truth value of True on at least one line of the truth table and a truth value of False on at least one line of the truth table.

3.

$(p \wedge \neg q) \vee (p \wedge q)$

$\equiv p \wedge (\neg q \vee q)$ by: distributive law

$\equiv p \wedge (q \vee \neg q)$ by: commutative law

$\equiv p \wedge T$ by: negation law

$\equiv p$ by: identity law

Therefore, $(p \wedge \neg q) \vee (p \wedge q) \equiv p$

The above solution to the problem represents the use of deductive reasoning to prove the logical equivalence. It is precisely this type of reasoning skill which is necessary for computer science students to master in order to excel at computer programming. Deductive reasoning is essential to design correct and good algorithms which are then translated into code.

In the study of sets, students also learn the power of formal systems. In set theory, students learn how to denote the members of a set using both a membership list and set-builder notation. They learn operations which can be performed on sets, such as union, intersection and set difference. They also apply the knowledge they learned in the section on logic to prove membership in a set expression by using membership tables and set identity laws.

Some problems the students are expected to solve include:

1. Let A= {1, 2, 3, 4, 5} and B = {0, 3, 6}
   a. Find $A \cup B$
   b. Find $A \cap B$
   c. Find $A - B$
   d. Find B - A
2. Prove $(A \cup B) - B = A - B$ using a membership table.
3. Prove $(A \cup B) - C = (A - C) \cup (B - C)$ using set identity laws.

In the section on matrices, students are expected to solve problems such as finding the sum or product of a pair of matrices, finding an identity matrix or using matrices to solve a system of linear equations. Some of the problems the students solve are listed below:

1. Find a matrix **A** such that:

$$\begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix} A = \begin{bmatrix} 3 & 0 \\ 1 & 2 \end{bmatrix}$$

2. Given:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

a. Find: $A \vee B$

b. Find: $A \wedge B$

c. Find: $A \odot B$

2. Instructors will help students understand the concept of proof as a chain of inferences. How will instructors help students understand this concept?

There are two ways that we address this issue in ICS-141. In the first case, we formally cover the logical foundations of mathematical reasoning including the concepts of axioms, rules of inference, lemmas, theorems, and corollaries. We expose students to the various tautologies of addition, simplification, and classical formal methods of proofs including modus ponens, modus tollens, hypothetical syllogism, and disjunctive syllogism, and include direct and indirect proofs and proof by contradiction. In the second case we introduce new theorems in mathematics by going through examples of their usage, then analyzing the fundamental principles in these examples, and then formally stating the theorems. The second step that extracts the fundamental principles is a very effective way of getting students to appreciate the concept of a proof as a chain of inferences. This kind of activity is carried out in all parts of the course.

An example of a formal proof covered in the section on logic, follows:

1. Suppose we have the following premises:
"It is not sunny and it is cold."
"We will swim only if it is sunny."
"If we do not swim, then we will canoe."
"If we canoe, then we will be home early."

Given these premises, prove the theorem
"We will be home early" using inference rules.

Let us adopt the following abbreviations:

sunny = **"It is sunny"**; *cold* = **"It is cold"**;
swim = **"We will swim"**; *canoe* = **"We will canoe"**; *early* = **"We will be home early"**.

Then, the premises can be written as:
(1) $\neg sunny \wedge cold$ (2) $swim \rightarrow sunny$
(3) $\neg swim \rightarrow canoe$ (4) $canoe \rightarrow early$

Step	Proved by
1. $\neg sunny \wedge cold$	Premise #1.
2. $\neg sunny$	Simplification of 1.
3. $swim \rightarrow sunny$	Premise #2.
4. $\neg swim$	Modus tollens on 2,3.
5. $\neg swim \rightarrow canoe$	Premise #3.
6. $canoe$	Modus ponens on 4,5.
7. $canoe \rightarrow early$	Premise #4.
8. $early$	Modus ponens on 6,7, which was to be shown.

Students identify argument forms, as in the following sample exercise.

2. What rule of inference is used in each of these arguments?
   a. Alice is a mathematics major. Therefore, Alice is either a mathematics major or a computer science major.
   b. Jerry is a mathematics major and a computer science major. Therefore, Jerry is a mathematics major.
   c. If it is rainy, then the pool will be closed. It is rainy. Therefore, the pool is closed.
   d. If there is a hurricane today, the university will close. The university is not closed. Therefore, there is not a hurricane today.
   e. If I go to the beach, then I will stay in the sun too long. If I stay in the sun too long, I will get a sunburn. Therefore, if I go to the beach, then I will get a sunburn.

When covering the section on elementary number theory, students practice writing direct proofs, indirect proofs and proofs by contradiction. For example, students may be asked to prove that the sum of two odd integers is even. A direct proof to solve this problem could be written as:

Let x and y be two odd integers. Then, by the definition of an odd integer there exists another integer j such that $x = 2j + 1$, and there exists another integer k such that $y = 2k + 1$. Thus, $x + y = 2j + 1 + 2k + 1 = 2j + 2k + 2 = 2(j + k + 1)$, which by definition is an even integer. Therefore, we have proved that the sum of odd integers x and y, is an even integer.

Further, in the sections on predicate logic and the use of quantifiers, students solve problems where they express English statements in symbols and translate symbols to English statements. They use the quantified expressions in writing proofs.

3. Instructors will teach students how to apply formal rules or algorithms. How will instructors meet this hallmark?

A major focus of this course is to expose students to the practical usage of mathematics. Students are required to solve a significant number of problems for homework assignments, quizzes and in-class exams. An inherent part of this effort is to apply formal rules of logic and to apply algorithmic computations to the problem-solving process. In elementary number theory we introduce the concept of congruences and ask students to relate this concept to examples like the representation of numbers in various bases, the use of bar codes in supermarkets, and their usage in ISBN codes to uniquely identify books.

Students are asked to trace through algorithms to verify that they produce the desired result. A number of searching (linear, binary) and sorting (insertion, selection, bubble) algorithms are tested. To find the greatest common divisor for the Euclidean Algorithm, quotients and remainders are traced and verified by the students. Students solve problems involving the conversion between bases.

Some problems the students are asked to solve include:

1. Show the contents of the array after each pass of the bubble sort algorithm for an array containing: 15, 52, 37, 29, 4, 46, 26

2. Use the Euclidean Algorithm to find the greatest common divisor of 414 and 662.

3. Convert the binary number 111011101111010 to its hexadecimal equivalent.

4. Convert the decimal number 83,904 to its binary equivalent.

5. Use the public key encryption function c = (p + k) mod 26 to encrypt the message: Hello World.

6. Use RSA encryption function $C = M^c \bmod n$, to encrypt the message: Test Over

4. Students will be required to use appropriate symbolic techniques in the context of problem solving, and in the presentation and critical evaluation of evidence. What symbolic techniques will be required and in what contexts? How will presentations and evaluations of evidence be incorporated into the course?

Symbolic techniques are most prominent in the logic and sets portions of the course. In logic, they are used to represent lemmas, theorems, and corollaries and the results of the use of rules of inference on them. In set theory they are used to represent sets and the results of basic operations on sets.

The examples relating to the study of symbolic logic, sets and matrices presented in the answer to hallmark one are equally applicable to this hallmark. In addition, symbolic techniques are used in the study of predicates and quantified statements. Students are required to translate between English statements and their symbolic representation. Further, the symbolic representations are used to develop proofs involving quantified statements. Some of the problems the students are asked to complete, include the following:

1. Let W(x,y) = "x has visited website y" The u.d. for x is all students at LCC and for y is all web sites. Express in English:

   a. W(Maria Santos, www.hawaii.edu)
   b. $\exists x$ W(x, www.fox.com)
   c. $\exists y$ W(John Au, y)
   d. $\exists y$ (W(John Au, y) $\wedge$ W(Maria Santos, y))
   e. $\exists y \forall z$ ((y $\neq$ John Au) $\wedge$ (W(John Au, z) $\rightarrow$ W(y, z)))
   f. $\exists x \exists y \forall z ((x \neq y) \wedge (W(x, z) \rightarrow W(y, z)))$

2. Let L(x,y) = "x loves y" and the u.d. for both x and y is all people. Express using quantifiers, logical connectives and L(x,y).

   a. No one loves John Stevens.
   b. Everybody loves somebody.
   c. Somebody loves everybody.
   d. Nobody loves everybody.
   e. Everyone loves him(her)self.

5. The course will not focus solely on computational skills. What reasoning skills will be taught in the course?

In the logic and mathematical reasoning portion of the course the emphasis will be on proof techniques as a reasoning skill. Students will be exposed to the idea of a rigorous argument to support a concept.

The answer provided to hallmark two provides some examples of the ways reasoning skills are taught in this course. That section describes the use of reasoning to provide that a proposition follows from one or more other propositions using logical inference rules. The use of set identity laws to prove membership in a set expression is another example presented there. Finally, writing direct and indirect proofs using the properties of integers was presented.

The problem, Prove $\overline{A \cap B} = \overline{A} \cup \overline{B}$ using set identity laws and set builder notation illustrates the development of deductive reasoning skills necessary for computer programmers.

$\overline{A \cap B}$	$= \{x \mid x \notin A \cap B\}$	by def. of complement
	$= \{x \mid \neg (x \in (A \cap B))\}$	by def. of $\notin$ symbol
	$= \{x \mid \neg (x \in A \wedge x \in B)\}$	by def. of intersection
	$= \{x \mid \neg (x \in A) \vee \neg (x \in B)\}$	by DeMorgan's law
	$= \{x \mid x \notin A \vee x \notin B\}$	by def. of $\notin$ symbol
	$= \{x \mid x \in \overline{A} \vee x \in \overline{B}\}$	by def. of complement
	$= \{x \mid x \in \overline{A} \cup \overline{B}\}$	by def. of union
	$= \overline{A} \cup \overline{B}$	by def. set builder notation

In addition to those already mentioned, a section on inductive reasoning techniques is presented and students write proofs using the principle of mathematical induction. In inductive reasoning, the student is asked to do two things: (a) prove an expression is true for some initial value and (b) prove the implication $P(k) \rightarrow P(k+1)$. When proving the implication, the student assumes P(k) is true and shows that P(k+1) follows from it. In the following sample problem, the Basis step fulfills requirement (a) and the Inductive step fulfills requirement (b). A significant amount of algebraic manipulation is often used in proving requirement (b), but the focus of an inductive proof is really on the methodology described above.

Write an inductive proof to show that if n is a positive integer, then

$$1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}.$$

Basis Step, Show that P(1) is true.

P(1) is true, because $1 = \frac{1(1+1)}{2}$.

Inductive Step, Assume that P(k) is true for some arbitrary integer k, and prove that P(k+1) is also true.

Prove that P(k+1) = 1 + 2 + 3 + … + k + (k+1) = $\frac{(k+1)(k+1+1)}{2} = \frac{(k+1)(k+2)}{2}$.

$$1 + 2 + 3 + \ldots + k + (k+1) = \frac{k(k+1)}{2} + (k+1)$$
$$= \frac{k(k+1) + 2(k+1)}{2}$$
$$= \frac{(k+1)(k+2)}{2}.$$

We have completed both the Basis step and the Inductive step, so by mathematical induction we have shown that 1 + 2 + 3 + … + n = $\frac{n(n+1)}{2}$, for all positive integers n.

A problem that the student is expected to solve is:

Use mathematical induction to prove: $1^2 + 2^2 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}$, for all positive integers n.

6. Instructors will build a bridge from theory to practice and show students how to traverse this bridge. How will instructors help students make connections between theory and practice?

There are several applications that can be made to computer science practices in this course. Students will see how algorithms can be implemented in programming languages. Students will see the direct application to data storage, hashing, and encryption techniques.

Students will implement some of the algorithms presented in the text in the syntax of a formal programming language and test the algorithms on the computer. Since the current programming language taught in the programming course ICS-111, is Java. Students implement some of the searching and sorting algorithms studied in Java and test them on the computer. Students can also implement the Euclidean Algorithm to find the greatest common divisor of two integers.

When studying the properties of integers and quotients and remainders, the students solve problems using encryption and decryption algorithms previously mentioned. The students also apply this concept to hashing functions, which convert a longer key value into a shorter value to facilitate ordered storing and retrieving of data. For example, when storing data about a student or employee, you might use their social security number or identification number. To create arrays using these values as indices would require you to reserve a lot of memory, perhaps depleting the available memory on a computer. A hashing function can be used to convert an index into an array for a particular number, allowing for fast storage and retrieval of the data without reserving more space than is necessary.

LEEWARD COMMUNITY COLLEGE
Mathematics and Natural Sciences Division
Course Syllabus
ICS 141 -- DISCRETE MATH FOR COMP SCI I (3.0 credits)

Instructor:

Office Hours:

Office Location:

Contact Information:

Catalog Course Description:
Includes logic, sets, functions, matrices, algorithmic concepts, mathematical reasoning, recursion, counting techniques, and probability theory. * (45 lecture hours)

Co-requisites:
None

Prerequisites:
MATH 103 College Algebra, or equivalent, or consent of instructor.

Recommended Preparations:
None

Required Textbook:
"Discrete Mathematics and Its Applications" 6th Edition by Kenneth H. Rosen.

Required Supply:
1 Scientific Calculator

Other Requirements:
- WebCT Account: Each student will be required to use WebCT. A WebCT account from UH-Manoa has been created for you. Instructions for accessing your account will be provided in class.

Student Learning Outcomes:
Upon completion of ICS-141, the student should be able to

Analyze issues and apply mathematical problem solving skills to plan courses of actions in decision-making situations, using:

1. Basic mathematical formal logic.
2. Proofs.
3. Recursion.
4. Analysis of algorithms.
5. Sets.
6. Combinatorics.
7. Relations.
8. Functions.
9. Matrices.
10. Probability.

Grading Policy:

- ✶ Quizzes: A total of five quizzes worth 25 points each will be given in the course. Your best four quiz scores contribute 100 points toward your final grade.

- ✶ Exams: There are two exams worth 50 points each. The first exam covers the first half of the course and the second exam covers the second half of the course. Exams are worth 100 points.

Final Grades will be based upon:

A	100-90%	200 - 180 points
B	89.9-80%	179 - 160 points
C	79.9-70%	159 - 140 points
D	69.9-60%	139 - 120 points
F	59.9 - 0%	119 - 0 points

Grading Policies:

1. No make-up exams will be given unless your absence is due to a medical emergency or is work related. Written verification from your doctor or employer is required before permission for a make-up exam is granted. No exceptions!!!

2. Quizzes must be completed by the date due.

Student Contributions:
Students are expected to spend at least one to two hours per class period in addition to class time on this course. Regular attendance and active participation in class activities and discussions are vital for success in this course.

Student with Disabilities Statement:

Students with documented disabilities who believe that they may need accommodations in this class are encouraged to contact the Coordinator of the KAKO‘O ‘IKE (KI) program as soon as possible to ensure that such accommodations are implemented in a timely fashion. The KI office is located in L-208, across from the elevator in the library building or call for information at 455-0421.

COURSE SYLLABUS
(subject to revision)

Topic/Homework

Sec. 1.1: Propositional Logic
Sec. 1.2: Propositional Equivalences
Sec. 1.3: Predicates & Quantifiers
Sec. 1.4: Nested Quantifiers
Sec. 1.5: Rules of Inferences
Sec. 1.6: Proof Methods & Strategies
Quiz One

Sec. 2.1: Sets
Sec. 2.2: Set Operations
Sec. 2.3: Functions
Sec. 2.4: Sequences and Summations
Sec. 3-1: Algorithms
Quiz Two

Sec. 3.2: The Growth of Functions
Sec. 3.3: Complexity of Algorithms
Sec. 3.6: Integers and Algorithms
Sec. 3.5: Primes and Greatest Common Divisors
Sec. 3.7: Applications of Number Theory
Quiz Three

Exam One

Sec. 3.8: Matrices
Sec. 4.1: Mathematical Induction
Sec. 4.3: Recursive Definitions
Sec. 4.4: Recursive Algorithms
Quiz Four

Sec. 5.1: The Basics of Counting
Sec. 5.2: The Pigeonhole Principle

Sec. 5.3: Permutations & Combinations
Sec. 5.4: Binomial Coefficients
Sec 6.1: Introduction to Discrete Probability
Quiz Five

Sec. 6.2: Probability Theory
Sec. 6.3: Bayes Theorem
Sec. 6.4: Expected Value & Variance

Exam Two