Chapter 4 roadmap:

- Subset proofs (week-before Friday)
- Set equality and emptiness proofs (last week Monday)
- Conditional and biconditional proofs (last week Wednesday)
- Proofs about powersets (last week Friday)
- Review for Test 2 (**today**)
- Test 2 (Wednesday)
- (Begin Chapter 5 Relations Friday)

Today:

- General review of Ch 3 & 4
- What to expect
- Specific warnings
- How can I help you?

Goals of this course

- Write programs in the functional style
- Think recursively
- Understand sets, relations, and functions so that they can model real-world (and abstract) information
- Use formal logic to prove mathematical propositions.

Concepts of the first two chapters

- The system of propositional logic, including logical equivalences and arguments
- Boolean operations and predicates in functional programming
- Quantification
- Proofs of set propositions
- Proofs of conditional propositions

Concepts	Testable skills
3.1. The definition of a *proposition*; propositional/logical/Boolean values; propositional forms. Propositional operators: negation, conjunction, and disjunction. The Python boolean operators `not`, `and`, and `or`. Boolean-valued Python functions, including the pattern of recursive functions with both a `True` and `False` base case. Python assertions.	Evaluate propositional expressions Write Python expressions and functions that use Boolean operators Write recursive, Boolean-valued Python functions
3.2. Truth tables for analyzing propositional forms and determining whether two propositional forms are logically equivalent. Verifying logical equivalence between propositional forms by applying known equivalences.	Verify logical equivalences using a truth table. Verify logical equivalences by applying known equivalences from Theorem 3.1. (Game 1)

Concepts	Testable skills
3.4. The logical *conditional operator*. Conditional expressions in Python (review). The negation, converse, inverse, and contrapositive of a conditional.	
3.5. Arguments and argument forms. Critical rows. The eight named argument forms: modus ponens, modus tollens, specialization, generalization, elimination, transitivity, division into cases, contradiction.	Verify argument forms using a truth table. Verify argument forms by applying known argument forms (and logical equivalences). (Game 2)

Concepts	Testable skills
3.6. The various definitions of (or perspectives on) the term *predicate*. The use of Python boolean functions as parameters to functions like `filter_set` and `filter_list`. Anonymous functions, also called lambda expressions.	Write Python functions that make use of lambda expressions along with given functions like `filter_set` or `filter_list`.
3.7. Universal and existential quantification, along with the symbols and notation for expressing quantified propositions. How to prove or disprove quantified propositions. Multiple quantification. Python built-in functions `all` and `any`.	Write Python functions whose specification involves quantification. Write Python functions that use `all` or `any`.

Concepts

3.8. The structure of proofs for propositions that are universally quantified, existentially quantified, or conditional. How to use division into cases in a structured proof. How to apply universally or existentially quantified proposition and definitions in a proof.

Testable skills

(*There will not be any Game 3 problems on the test... but the lessons from Game 3 should be applicable in writing proofs from Chapter 4.*)

Concepts	**Testable skills**
4.1. The definition of theorem, in the context of propositions, conjectures, and axioms. The general structure of a proof: supposition, justifying each step, "therefore" etc. The element argument for proving subset propositions (Set Form 1). How to use the definitions of union, intersection, difference, symmetric difference, complement, and Cartesian product in a proof. Various proof elements like division into cases, substitution, etc.	Write proofs of subset propositions.
4.2. Two applications of the element argument for proving set equality propositions (Set Form 2).	Write proofs of set-equality propositions.
4.3. Proof-by-contradiction for proving set-emptiness propositions (Set Form 3).	Write proofs of set-emptiness propositions.

Concepts	Testable skills
4.4 Conditional and biconditional proofs.	Write proofs of conditional and biconditional propositions.
4.5. Number theory proofs. The formal definition of even and odd. The definition of "divides" ($\mid$).	Write proofs of propositions about numbers.
4.6. How to use the definition of powerset in a proof.	Write proofs of propositions about powersets.

Which of the following are true?

$$
\begin{aligned}
-((x-y)+(x-z)) &= -(x-y)-(x-z) \\
-((x-y)+(x-z))\cdot z &= -(x-y)-(x-z)\cdot z \\
\sim(p \wedge q) &\equiv \sim p \vee \sim q \\
\sim(p \wedge q) \wedge r &\equiv \sim p \vee \sim q \wedge r
\end{aligned}
$$

Which of the following are true?

$$
\begin{aligned}
(x+y)+z &= x+(y+z) \\
(x-y)+z &= x-(y+z) \\
(p \vee q) \vee r &\equiv p \vee (q \vee r) \\
(p \vee q) \wedge r &\equiv p \vee (q \wedge r)
\end{aligned}
$$

$$
\begin{aligned}
&((q \wedge (p \wedge (p \vee q))) \vee (q \wedge \sim p)) \wedge \sim q & \\
\equiv\quad & ((q \wedge p) \vee (q \wedge \sim p)) \wedge \sim q & \text{Absorption} \\
\equiv\quad & (q \wedge (p \vee \sim p)) \wedge \sim q & \text{Distributivity} \\
\equiv\quad & (q \wedge T) \wedge \sim q & \text{Negation} \\
\equiv\quad & q \wedge \sim q & \text{Identity} \\
\equiv\quad & F & \text{Negation}
\end{aligned}
$$

WRONG!

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

$$\equiv \quad ((q \wedge p) \vee (q \wedge \sim p)) \wedge \sim q \qquad \text{Absorption}$$

$$\equiv \quad (q \wedge p) \vee ((q \wedge \sim p) \wedge \sim q) \qquad \text{Associativity}$$

Suppose we were to show that $\sim(\sim p \wedge q) \vee (p \vee \sim p) \equiv p \vee \sim q$.

Do this:

$$
\begin{array}{lll}
\sim(\sim p \wedge q) \vee (p \wedge \sim p) & & \\
\equiv & \sim(\sim p \wedge q) \vee F & \text{by negation law} \\
\equiv & \sim(\sim p \wedge q) & \text{by identity law} \\
\equiv & p \vee \sim q & \text{by De Morgan's}
\end{array}
$$

Don't do this:

$$
\begin{array}{rcll}
\sim(\sim p \wedge q) \vee (p \wedge \sim p) & \equiv & p \vee \sim q & \\
\sim(\sim p \wedge q) \vee F & \equiv & p \vee \sim q & \text{by negation law} \\
\sim(\sim p \wedge q) & \equiv & p \vee \sim q & \text{by identity law} \\
p \vee \sim q & \equiv & p \vee \sim q & \text{by De Morgan's}
\end{array}
$$

Modus Ponens

$p \to q$

p

$\therefore q$

Modus Tollens

$p \to q$

$\sim q$

$\therefore \sim p$

Generalization

p

$\therefore p \vee q$

Specialization

$p \wedge q$

$\therefore p$

Elimination

$p \vee q$

$\sim p$

$\therefore q$

Transitivity

$p \to q$

$q \to r$

$\therefore p \to r$

Division into cases

$p \vee q$

$p \to r$

$q \to r$

$\therefore r$

Contradiction

$p \to F$

$\therefore \sim p$

For next time:

Study for test...

Read Sections 5.(1 & 2) for Friday, Mar 7
Take quiz