Tandon Bridge| Discrete Math

- 1. Logic
  - 1.1 Propositions and Logical Operations
    - Conjunction operation: $p \land q$ => "p and q"
    - Disjunction operation: $p \lor q$ => "p or q"

      > The **exclusive or** (usually denoted with the symbol ⊕) of p and q evaluates to true when p is true and q is false or when q is true and p is false.
      > The **inclusive or** operation is the same as the disjunction (∨) operation and evaluates to true when one or both of the propositions are true. (Since the inclusive or is most common in logic, it is just called "or" for short.)

    - Negation operation: the negation of proposition p is $\neg p$ => "not p"
  - 1.2 Compound Propositions:
    - Order of operations in absence of parentheses: ¬ (not)-> ∧ (and) -> ∨ (or)
  - 1.3 Conditional Statements: $p \to q$ => "if p then q" ($p \to q$ is false if p is true and q is false)

p	q	p → q
T	T	T
T	F	F
F	T	T
F	F	T

    - English expressions of the conditional operation:

Consider the propositions:

rainbow rain

$p \to q$

p: You mow Mr. Smith's lawn.
q: Mr. Smith will pay you.

If p, then q.	If you mow Mr. Smith's lawn, then he will pay you.
If p, q.	If you mow Mr. Smith's lawn, he will pay you.
q if p	Mr. Smith will pay you if you mow his lawn.
p implies q.	Mowing Mr. Smith's lawn implies that he will pay you.
p only if q.	You will mow Mr. Smith's lawn only if he pays you.
p is sufficient for q.	Mowing Mr. Smith's lawn is sufficient for him to pay you.
q is necessary for p.	Mr. Smith's paying you is necessary for you to mow his lawn.

- Converse($p \to q$ is $q \to p$), Contrapositive($\neg q \to \neg p$), Inverse($\neg p \to \neg q$)

p	q	¬q→¬p
T	T	T
T	F	F
F	T	T
F	F	T

- Biconditional operation: "p if and only if q" => $p \leftrightarrow q$ (is true when p and q have the same truth value)

p	q	p ↔ q
T	T	T
T	F	F
F	T	F
F	F	T

p	q	¬q↔¬p
T	T	T
T	F	F
F	T	F
F	F	T

"p is necessary and sufficient for q" or "if p then q, and conversely"
The term iff is an abbreviation of the expression "if and only if", as in "p iff q"

- Compound propositions with conditional and biconditional operations: ( ); $\wedge$, $\vee$, $\neg$; $\to$ / $\leftrightarrow$

- 1.4 Logical Equivalence:
  - A compound proposition is a **tautology**/**contradiction** if the proposition is always **true**/**false**.
  - Two compound propositions are logically equivalent if they have the same truth value regardless of the truth values of their individual propositions.
    - Show logical equivalence using truth tables:
      - $p \to \neg p \equiv \neg p$ $\quad \neg(p \vee q) \equiv (\neg p \wedge \neg q)$ $\quad \neg(p \wedge q) \equiv (\neg p \vee \neg q)$

- 1.5 Laws of propositional logic: Substituting logically equivalent propositions. $p \to q \equiv \neg p \vee q$

Table 1.5.1: Laws of propositional logic.

1.	Idempotent laws:	$p \vee p \equiv p$	$p \wedge p \equiv p$
2.	Associative laws:	$(p \vee q) \vee r \equiv p \vee (q \vee r)$	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
3.	Commutative laws:	$p \vee q \equiv q \vee p$	$p \wedge q \equiv q \wedge p$
4.	Distributive laws:	$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$
5.	Identity laws:	$p \vee F \equiv p$	$p \wedge T \equiv p$
6.	Domination laws:	$p \wedge F \equiv F$	$p \vee T \equiv T$
7.	Double negation law:	$\neg\neg p \equiv p$	
8.	Complement laws:	$p \wedge \neg p \equiv F$ $\neg T \equiv F$	$p \vee \neg p \equiv T$ $\neg F \equiv T$
9.	De Morgan's laws:	$\neg(p \vee q) \equiv \neg p \wedge \neg q$	$\neg(p \wedge q) \equiv \neg p \vee \neg q$
10.	Absorption laws:	$p \vee (p \wedge q) \equiv p$	$p \wedge (p \vee q) \equiv p$
11.	Conditional identities:	$p \to q \equiv \neg p \vee q$	$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$

- 1.6 Predicate Logic and Quantifiers
  - Predicate: A logical statement whose truth value is a function of one or more variables(contains a variable)
  - Universal quantifier
    - If all the variables in a predicate are assigned specific values from their domains, then the predicate becomes a proposition with a well defined truth value. Or use a quantifier :

      > **∀x P(x)**: "for all x, P(x)" or "for every x, P(x)" => P(x) is true for every possible value for x in its domain

      - The symbol ∀ is a universal quantifier
      - The statement ∀x P(x) is called a universally quantified statement.
      - ∀x P(x) is a proposition. ∀x P(x) is true if and only if P(n) is true for every n in the domain.
  - Existential quantifier
    - The statement ∃x P(x) asserts that P(x) is true for at least one possible value for x in its domain

      > **∃x P(x)** : "There exists an x, such that P(x)"

      - The symbol ∃ is an existential quantifier
      - The statement ∃x P(x) is called an existentially quantified statement.
      - ∃x P(x) is a proposition because it is either true or false. ∃x P(x) is true if and only if P(n) is true for at least one value n in the domain of variable x.
- 1.7 Quantified statements

- The universal and existential quantifiers are generically called **quantifiers**. A logical statement that includes a universal or existential quantifier is called a quantified statement. The quantifiers ∀ and ∃ are applied before the logical operations (∧, ∨, →, and ↔) used for propositions.
- A variable x in the predicate P(x) is a **free variable** because the variable is free to take on any value in the domain. The variable x in the statement ∀x P(x) is a **bound variable** because the variable is bound to a quantifier. A statement with no free variables is a proposition because its truth value can be determined.

- 1.8 De Morgan's law for quantified statements
  - **¬∀x F(x) ≡ ∃x ¬F(x)**

    Figure 1.8.1: De Morgan's law for universally quantified statements.

    Domain of discourse = $\{a_1, a_2, \ldots, a_n\}$

    $$\neg\forall x\, P(x) \equiv \exists x\, \neg P(x)$$

    $$\neg(P(a_1) \land P(a_2) \land \ldots \land P(a_n)) \equiv \neg P(a_1) \lor \neg P(a_2) \lor \ldots \lor \neg P(a_n)$$

  - **¬∃x P(x) ≡ ∀x ¬P(x)**

    Figure 1.8.2: De Morgan's law for existentially quantified statements.

    Domain of discourse = $\{a_1, a_2, \ldots, a_n\}$

    $$\neg\exists x\, P(x) \equiv \forall x\, \neg P(x)$$

    $$\neg(P(a_1) \lor P(a_2) \lor \ldots \lor P(a_n)) \equiv \neg P(a_1) \land \neg P(a_2) \land \ldots \land \neg P(a_n)$$

- 1.9 Nested quantifiers
  - A logical expression with more than one quantifier that bind different variables in the same predicate is said to have nested quantifiers
  - Nested quantifiers of the same type: ∀x ∀y M(x, y)   ∃x ∃y M(x, y)
  - Alternating nested quantifiers: ∃x ∀y M(x, y)   ∀x ∃y M(x, y)

Table 1.9.2: De Morgan's laws for nested quantified statements.

$\neg\forall x\, \forall y\, P(x, y) \equiv \exists x\, \exists y\, \neg P(x, y)$	
$\neg\forall x\, \exists y\, P(x, y) \equiv \exists x\, \forall y\, \neg P(x, y)$	
$\neg\exists x\, \forall y\, P(x, y) \equiv \forall x\, \exists y\, \neg P(x, y)$	
$\neg\exists x\, \exists y\, P(x, y) \equiv \forall x\, \forall y\, \neg P(x, y)$	

- 1.10 More nested quantified statements
  - Using logic to express "everyone else": $(x \neq y) \rightarrow M(x, y)$
  - Expressing uniqueness in quantified statements: $\exists x(L(x) \land \forall y((x \neq y) \rightarrow \neg L(y)))$
  - Moving quantifiers in logical statements: $\forall x\, (A(x) \rightarrow \exists y\, M(x, y)) = \forall x\, \exists y (A(x) \rightarrow M(x, y))$
- 1.11 Logical Reasoning
  - hypotheses $\therefore$ conclusion (An argument is valid if the conclusion is true whenever the hypotheses are all true, otherwise the argument is invalid)
- 1.12 Rules of Inference with Propositions
  - Rules of inference known to be valid arguments

Rule of inference	Name
p $p \to q$ $\therefore q$	Modus ponens
$\neg q$ $p \to q$ $\therefore \neg p$	Modus tollens
p $\therefore p \lor q$	Addition
$p \land q$ $\therefore p$	Simplification
p q $\therefore p \land q$	Conjunction
$p \to q$ $q \to r$ $\therefore p \to r$	Hypothetical syllogism
$p \lor q$ $\neg p$ $\therefore q$	Disjunctive syllogism
$p \lor q$ $\neg p \lor r$ $\therefore q \lor r$	Resolution

- 1.13 Rules of inference with quantifiers
  - A value that can be plugged in for variable x is called an element of the domain of x
  - An **arbitrary** element of a domain has no special properties other than those shared by all the elements of the domain.
  - A **particular** element of the domain may have properties that are not shared by all the elements of the domain.

- The rules **existential instantiation** and **universal instantiation** replace a quantified variable with an element of the domain. The rules **existential generalization** and **universal generalization** replace an element of the domain with a quantified variable.

Table 1.13.1: Rules of inference for quantified statements.

Rule of Inference	Name	Example
c is an element (arbitrary or particular) ∀x P(x) ∴ P(c)	Universal instantiation	Sam is a student in the class. Every student in the class completed the assignment. Therefore, Sam completed his assignment.
c is an arbitrary element P(c) ∴ ∀x P(x)	Universal generalization	Let c be an arbitrary integer. $c \le c^2$ Therefore, every integer is less than or equal to its square.
∃x P(x) ∴ (c is a particular element) ∧ P(c)	Existential instantiation*	There is an integer that is equal to its square. Therefore, $c^2 = c$, for some integer c.
c is an element (arbitrary or particular) P(c) ∴ ∃x P(x)	Existential generalization	Sam is a particular student in the class. Sam completed the assignment. Therefore, there is a student in the class who completed the assignment.

- 2. Proof
  - 2.1 Introduction to proofs
    - A theorem is a statement that can be proven to be true.
    - Proof by Exhaustion: If the domain of a universal statement is small, it may be easiest to prove the statement by checking each element individually.
    - Counterexamples: an assignment of values to variables that shows that a universal statement is false.
  - 2.2 Direct proofs
    - In a direct proof of a conditional statement, the hypothesis p is assumed to be true and the conclusion c is proven as a direct result of the assumption.
    - A rational number, is defined to be a number that can be expressed as the ratio of two integers in which the denominator is non-zero.
  - 2.3 Proof by contrapositive
    - A **proof by contrapositive** proves a conditional theorem of the form $p \rightarrow c$ by showing that the contrapositive $\neg c \rightarrow \neg p$ is true. In other words, $\neg c$ is assumed to be true and $\neg p$ is proven as a result of $\neg c$.
  - 2.4 Proof by contradiction
    - A **proof by contradiction** starts by assuming that the theorem is false and then shows that some logical inconsistency arises as a result of this assumption. (indirect proof)
  - 2.5 Proof by cases
    - A **proof by cases** of a universal statement such as ∀x P(x) breaks the domain for the variable x into different classes and gives a different proof for each class. Every value in the domain must be included in at least one class.

- 3. Sets
  - 3.1 Sets and subsets
    - A set is an unsorted collection of objects. The objects in a set are called elements. A set may contain elements of different varieties
    - **roster notation**: a list of the elements enclosed in curly braces with the individual elements separated by commas

      > A = { 2, 4, 6, 10 }
    - **set builder notation**: specifying that the set includes all elements in a larger set that also satisfy certain conditions.

      > A = { x ∈ S : P(x) } "all x in S such that P(x)"
    - A finite set has a finite number of elements. An infinite set has an infinite number of elements. The cardinality of a finite set A, denoted by |A|
    - The universal set: usually denoted by the variable *U*, is a set that contains all elements mentioned in a particular context.
    - The set with no elements is called the empty set and is denoted by the symbol ∅. (null set { })
    - Mathematical sets

      > Table 3.1.1: Common mathematical sets.
      >
      > | Set | Symbol | Examples of elements |
      > |---|---|---|
      > | ***N*** is the set of ***natural numbers***, which includes all integers greater than or equal to 0. | **N** | 0, 1, 2, ... |
      > | ***Z*** is the set of all integers. | **Z** | ..., -2, -1, 0, 1, 2, ... |
      > | ***Q*** is the set of ***rational numbers***, which includes all real numbers that can be expressed as a/b, where a and b are integers and b ≠ 0. | **Q** | 0, 1/2, 5.23, -5/3 |
      > | ***R*** is the set of real numbers. | **R** | 0, 1/2, 5.23, -5/3, π, $\sqrt{2}$ |
  - 3.2 Sets of sets
    - The elements of a set are themselves sets.

      > The set A has four elements: { 1, 2 }, ∅, { 1, 2, 3 }, and { 1 }. For example, { 1, 2 } ∈ A. Note that 1 is not an element of A, so 1 ∉ A, although { 1 } ∈ A. Furthermore, { 1 } ⊈ A since 1 ∉ A.
    - The **power set** of a set A, denoted P(A), is the set of all subsets of A.

      > If A = { 1, 2, 3 }, then: P(A) = { ∅, { 1 }, { 2 }, { 3 }, { 1, 2 }, { 1, 3 }, { 2, 3 }, { 1, 2, 3 } }
    - The cardinality of the power set of A is 2^n, or |P(A)|=2^n.
  - 3.3 Union and intersection
    - The **intersection** of A and B, denoted A ∩ B and read "A intersect B": the set of all elements that are elements of both A and B
    - The **union** of two sets, A and B, denoted A ∪ B and read "A union B", is the set of all elements that are elements of A or B. (inclusive or)
  - 3.4 More set operations

- The **difference** between two sets A and B, denoted A - B: the set of elements that are in A but not in B. (not commutative)
- The **symmetric difference** between two sets, A and B, denoted $A \oplus B$, is the set of elements that are a member of exactly one of A and B, but not both. (commutative) $A \oplus B = (A - B) \cup (B - A)$

Operation	Notation	Description
Intersection	$A \cap B$	$\{ x : x \in A \text{ and } x \in B \}$
Union	$A \cup B$	$\{ x : x \in A \text{ or } x \in B \text{ or both} \}$
Difference	$A - B$	$\{ x : x \in A \text{ and } x \notin B \}$
Symmetric difference	$A \oplus B$	$\{ x : x \in A - B \text{ or } x \in B - A \}$
Complement	$\overline{A}$	$\{ x : x \notin A \}$

- 3.5 Cartesian products
  - The Cartesian product of A and B, denoted A x B: the set of all ordered pairs in which the first entry is in A and the second entry is in B. $A \times B = \{ (a, b) : a \in A \text{ and } b \in B \}$
    - The order of the elements in a pair is significant: If A and B are finite sets, then $|A \times B| = |A| \cdot |B|$.
  - ordered n-tuple: $A1 \times A2 \times \dots \times An = \{ (a1, a2, \dots , an) : ai \in Ai \text{ for all } i \text{ such that } 1 \le i \le n \}$
  - The Cartesian product of a set A with itself can be denoted as $A \times A$ or A^2 (A^k k times)
  - **R**^n, which is the set of all ordered n-tuples of real numbers. When n = 2, **R**^2 is the set of all pairs (x, y) such that x and y are real numbers.
  - A sequence of characters is called a **string**. The length of a string is the number of characters in the string.
    - A <u>binary string</u> is a string whose alphabet is {0, 1}
    - The <u>empty string</u> is the unique string whose length is 0 and is usually denoted by the symbol λ. Since {0, 1}0 is the set of all binary strings of length 0, {0, 1}0 = {λ}.
    - If s and t are two strings, then the **concatenation** of s and t (denoted st) is a longer string obtained by putting s and t together. Concatenating any string x with the empty string gives back x: xλ = x.
- 3.6 Set identities
  - All the elements are assumed to be contained in a universal set *U*.

$$x \in A \cap B \leftrightarrow (x \in A) \wedge (x \in B)$$

$$x \in A \cup B \leftrightarrow (x \in A) \vee (x \in B)$$

$$x \in \overline{A} \leftrightarrow \neg(x \in A)$$

  - The sets *U* and ∅ correspond to the constants true (T) and false (F):

$x \in \varnothing \leftrightarrow F$

$x \in U \leftrightarrow T$

- A **set identity** is an equation involving sets that is true regardless of the contents of the sets in the expression.

Table 3.6.1: Set identities.

Name	Identities	
Idempotent laws	$A \cup A = A$	$A \cap A = A$
Associative laws	$(A \cup B) \cup C = A \cup (B \cup C)$	$(A \cap B) \cap C = A \cap (B \cap C)$
Commutative laws	$A \cup B = B \cup A$	$A \cap B = B \cap A$
Distributive laws	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
Identity laws	$A \cup \varnothing = A$	$A \cap U = A$
Domination laws	$A \cap \varnothing = \varnothing$	$A \cup U = U$
Double Complement law	$\overline{\overline{A}} = A$	
Complement laws	$A \cap \overline{A} = \varnothing$ $\overline{U} = \varnothing$	$A \cup \overline{A} = U$ $\overline{\varnothing} = U$
De Morgan's laws	$\overline{A \cup B} = \overline{A} \cap \overline{B}$	$\overline{A \cap B} = \overline{A} \cup \overline{B}$
Absorption laws	$A \cup (A \cap B) = A$	$A \cap (A \cup B) = A$

- 3.7 Partitions
  - A partition of a non-empty set A is a collection of non-empty subsets of A such that each element of A is in exactly one of the subsets.

    For all i, $A_i \subseteq A$.
    For all i, $A_i \neq \varnothing$
    $A_1, A_2, \ldots, A_n$ are pairwise disjoint.
    $A = A_1 \cup A_2 \cup \ldots \cup A_n$

-

- 4. Functions
  - 4.1 Definition of functions
    - A function f that maps elements of a set X to elements of a set Y, is a subset of $X \times Y$ such that for every $x \in X$, there is *exactly one* $y \in Y$ for which $(x, y) \in f$.
    - For function $f: X \to Y$, an element y is in the range of f if and only if there is an $x \in X$ such that $(x, y) \in f$. Expressed in set notation: Range of f = $\{ y: (x, y) \in f, \text{ for some } x \in X \}$
    - Recognizing well-defined functions from arrow diagrams: *Every element in the domain is mapped to exactly one element in the target.*

- 4.2 Properties of functions
  - A function $f: X \to Y$ is **one-to-one** or **injective** if $x1 \neq x2$ implies that $f(x1) \neq f(x2)$. That is, f maps different elements in X to different elements in Y.
  - A function $f: X \to Y$ is **onto** or **surjective** if the range of f is equal to the target Y. That is, for every $y \in Y$, there is an $x \in X$ such that $f(x) = y$.
  - A function is **bijective** if it is both one-to-one and onto. A bijective function is called a bijection. A bijection is also called a one-to-one correspondence.
    - If $f: D \to T$ is onto, then for every element in the target, there is at least one element in the domain: $|D| \geq |T|$.
    - If $f: D \to T$ is one-to-one, then for every element in the domain, there is at least one element in the target: $|D| \leq |T|$.
    - If $f: D \to T$ is a bijection, then f is one-to-one and onto: $|D| \leq |T|$ and $|D| \geq |T|$, which implies that $|D| = |T|$.
- 4.3 The inverse of a function
  - If a function $f: X \to Y$ is a bijection, then the **inverse** of f is obtained by exchanging the first and second entries in each pair in f. The inverse of f is denoted by f ^-1 = { (y, x) : (x, y) ∈ f }.
  - A function $f: X \to Y$ has an inverse if and only if reversing each pair in f results in a well-defined function from Y to X. A function f has an inverse if and only if f is a bijection.
  - If f is a bijection from X to Y, then for every $x \in X$ and $y \in Y$, $f(x) = y$ if and only if f^-1(y) = x. Therefore the value of f^-1(y) is the unique element $x \in X$ such that $f(x) = y$. If f^-1 is the inverse of function f, then for every element $x \in X$, f^-1(f(x)) = x.
- 4.4 Composition of functions
  - f and g are two functions, where $f: X \to Y$ and $g: Y \to Z$. The **composition** of g with f, denoted g o f, is the function (g o f): $X \to Z$, such that for all $x \in X$, (g o f)(x) = g(f(x)).
  - Composition is associative, so the order in which one composes the functions does not matter: f o g o h = (f o g) o h = f o (g o h) = f(g(h(x)))
  - The i**dentity function** always maps a set onto itself and maps every element onto itself. The identity function on A, denoted IA: $A \to A$, is defined as IA(a) = a, for all $a \in A$. If a function f from A to B has an inverse, then f composed with its inverse is the identity function. If f(a) = b, then f^-1(b) = a, and (f^-1 o f)(a) =f-1(f(a)) = f-1(b) = a. Let $f: A \to B$ be a bijection. Then f^-1 o f = IA and f o f^-1 = IB.