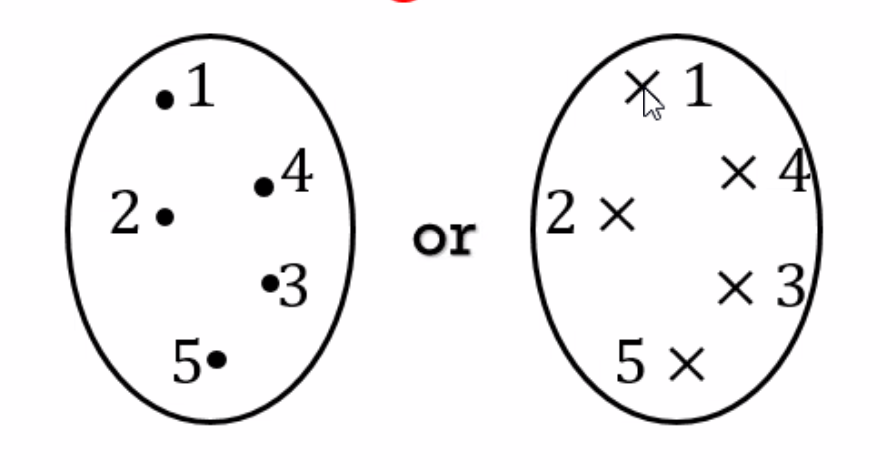
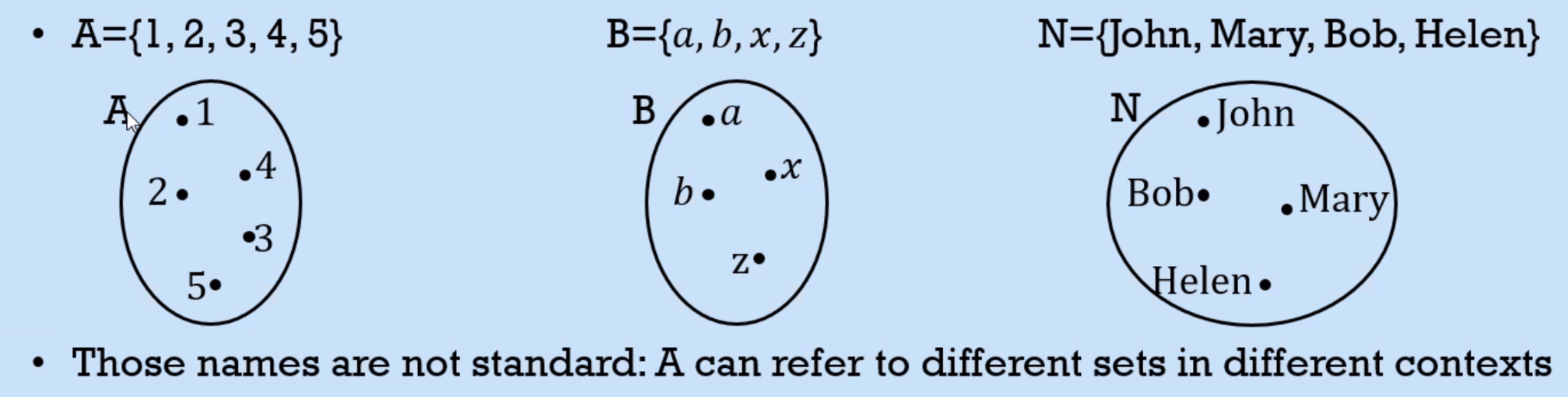
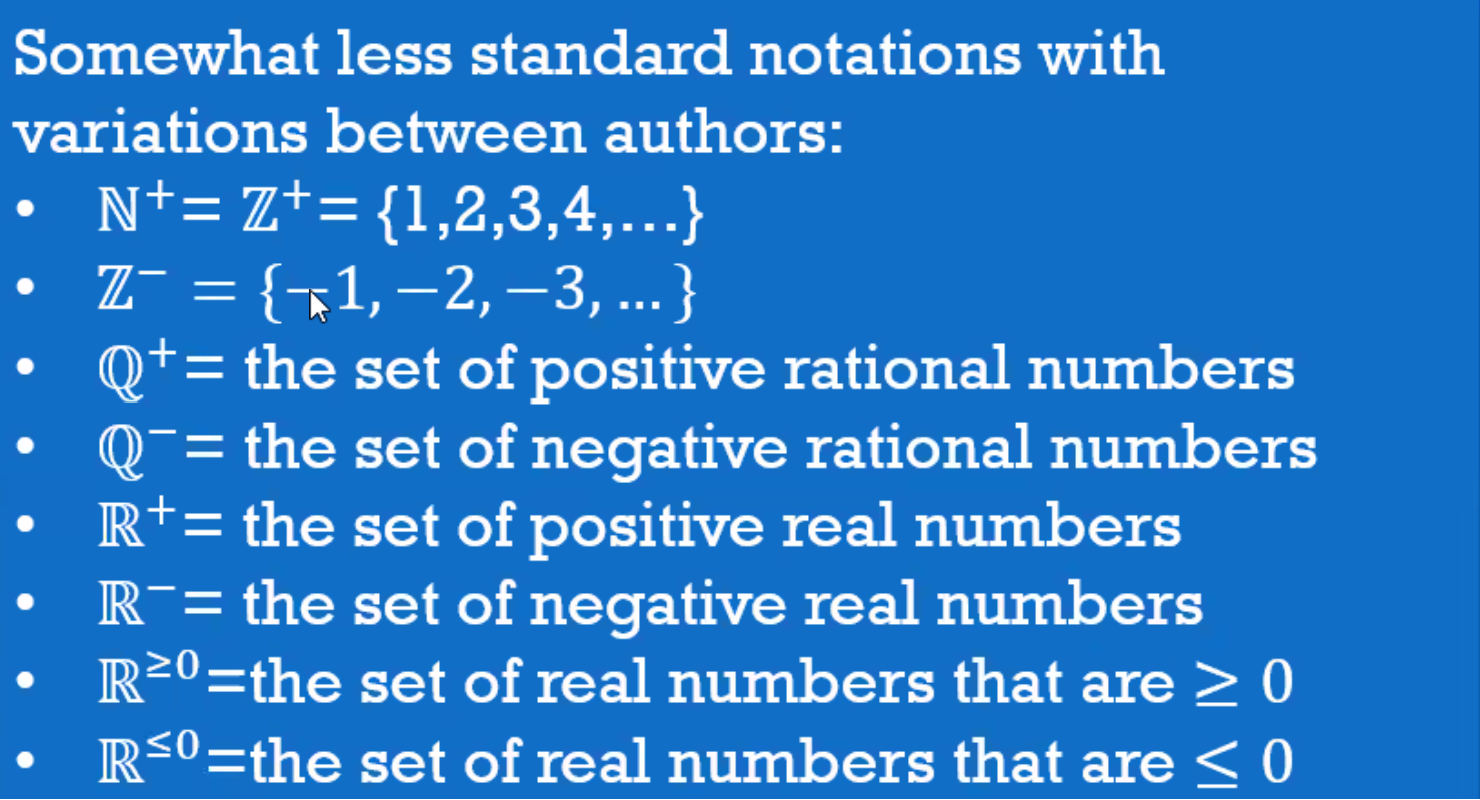
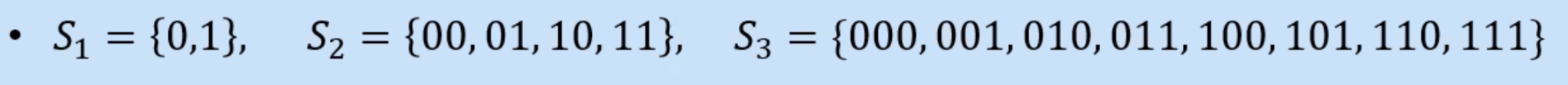
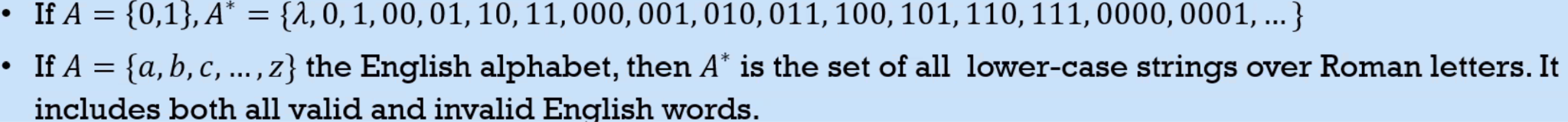
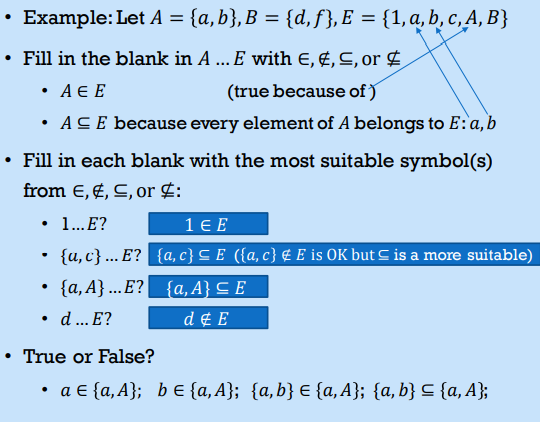
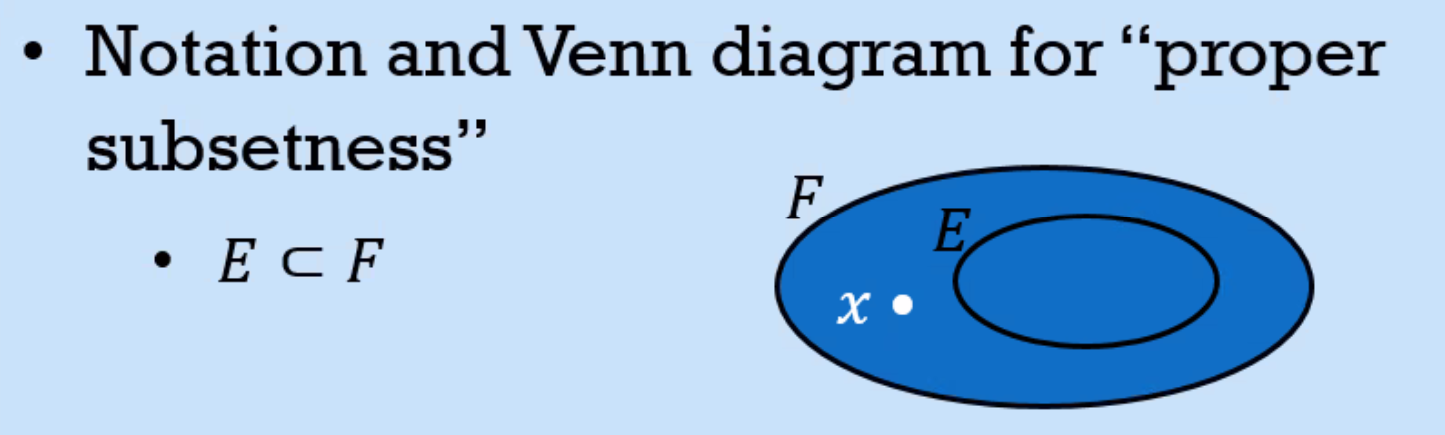
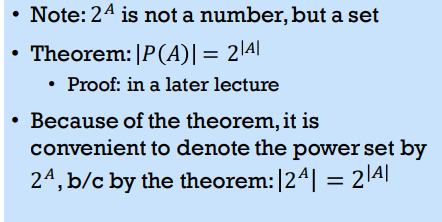
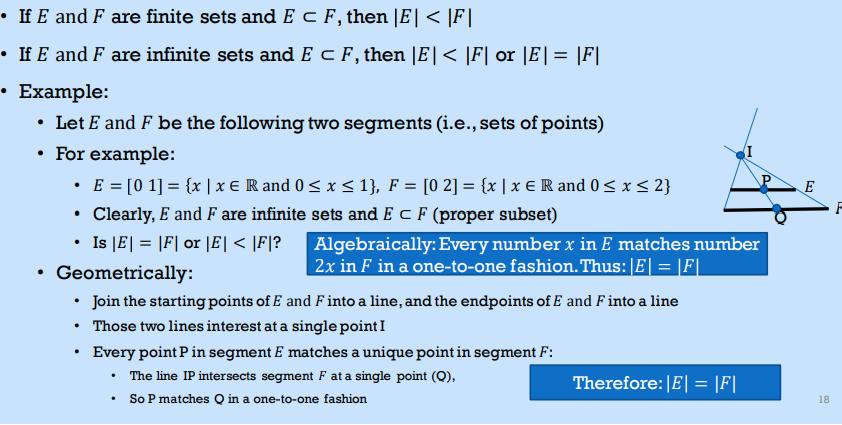
1/16/2024

## Set Theory Foundations part I

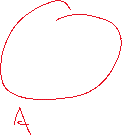
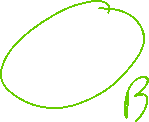
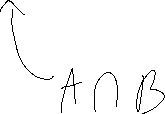
* Describe sets, subsets, super sets, belonging, and inclusion
* Represent sents in different
* Definition: A set is, intuitively, a collection of non-repeating objects (or items)
  + The objects in a set are called the elements (or members) of the set
  + Examples:
    - {a, 2, 3} is a set of 3 elements, namely, a, 2,3
    - {1}, {2,5,10, 20}
    - {0,2,4,6,8,10, …} = the set of all non-negative even integers
  + Observations
    - The elements are enclosed between curly braces { and }
    - The order of the elements is irrelevant
    - When the elements are listed explicitly, they are separated by commas
  + Types of set representations
    - Explicit/enumerative: the elements are listed out one by one. Ex: {1,2,3,4,5}
    - Implicit:
      * The elements are specified by a logical property/condidiotn/statemnt
      * Notation:{x | x is an integer and 1<=x<=5}
        + The above ste is the same as {1,2,3,4,5}
    - Venn diagrams:
      * 
  + Examples of sets:
    - { x | x is a prime number} = {2, 3, 5, 7, 11, 13, 17, 19, …}
    - {x | x is a word in the English Language} = the set of all valid English words
  + It is more convenient to give short-hand names to sets
    - Typically, we give uppercase Roman letter names
      * 
  + Standard Sets:
    - ℕ = {0,1,2,3,...}, which is the set of natural numbers
    - ℤ = {...,-2,-1,0,1,...}, integers. toInject asdfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff.
    - ℚ = {p/q | p and q are integers and q != 0} Rational numbers
    - ℝ = Real numbers = {−1, 3,5.7, 1 3 , 2, 𝜋𝜋, … }
    - ℂ = complex numbers
    - 
  + Other Useful sets
    - Binary strings are strings of 0s and 1s
      * For examples 00, 01, 11, are 2 bit strings 000 and 101 are 3 bit strings
    - For every positive integer n, let Sn = {x | x is an n-bit binary string}
      * 
    - Let A be a finite set (i.e., whose elements can be counted in finite time). A string over A is any sequence of concatenated elements of A. In that case, we call A an “alphabet”, and its elements “letters”
      * Example: A = {a,b,c}, then ab and acbabbc are two strings over A. Also, a,b, aaa are strings
      * Empty string: much as we have number 0-, we have emplyt string ε (epsilon) or sometimes Λ or λ(lambda)
    - The set of all strings over A is denoted A\*, i.e., A\* = {s | s is a sting over A}
      * 
      * A+ is A\* but without the empty string
* Belonging
  + Let E be a set
  + An object x is either an element of E or not
  + If x is an element of E:
    - We write x ∈ E
    - We read it as: x belongs to E
      * Or is an element of
  + If x is Not an element of E:
    - We write x ∉ E
    - We read it as : x does not belong to E
      * X is not an element of E
* Inclusion, Substes ⊆, supersets⊇
  + Let *E* be a set and *F* another set
  + Def: E is a subset of F (or E is included in F) if every element of E is an element of F
  + Notation: if E is a subset of F,
    - We write *E* ⊆F
    - We read it as “*E* is a subset of *F*” or “*E* is included in *F”I*
    - Imagine venn diagram
  + ℕ ⊆ℤ ⊆ℚ ⊆ℝ
  + **Def:**F is a superset of E
    - Notation F ⊇ E
  + **Def**: *E* is not a superset of F if ther is an element in E that does not belong to F
    - Notation ⊈ F
* Fine point s of belonging and inclusion
  + The members of a set can be “atomic” elements or sets in their right
* Therefore, a set can be either
  + An element of another set, or
  + A subset of another set, or
  + Both, or
  + Neither
  + Example:
    - 
* Set Equality and inequality
  + **Def** tow sets A and B are said to be equal (A=B) if every element of A is an element of B and vice versa.
  + Equivantly 𝐴 = 𝐵 if 𝐴 ⊆ 𝐵 and 𝐵 ⊆ *A*
  + Def: Two sets A and B are said to be different (𝐴 ≠ 𝐵) if one of the sets has an element that is not in the other set
* Set Cardinality, Finite sets, Infinite sets
  + Definition: The cardinality of a set is number of elements in the set
  + Notation: The cardinality of a set *A* is detonated |*A*|
  + A set A is said to be finite if its cardinality is a (finite) integer. That is, its elements can be counted in a finite amount of time.
  + A set A is said to be infinite if it is not countable in a finite amount of time
    - If A is infinite, Then |A| = ∞
  + Examples:
    - If *A =* { 0, 2,4,6,8}, |A| = 5
* Proper Subset ⊂
  + Def: a set *E* is called a proper subset of a set F if the fololowijng two conditions are satisfied:
    - E⊆F and
    - E != F, that is F has an element x that does not belong to E
  + 
* Definition: An empty ste is a set that has no elements
  + Notes:
    - An empty set is equivalent to the “void”
    - An empty set what 0 (zero) is to numbers
  + Theorem: There can’t be multiple empty sets. That is, there is only empty set.
  + Proof: By contradiction.
    - Assume there are (at least) two different empty sets ∅1 and ∅2
    - ∅1 ≠ ∅2 implies that one of those two sets has an element that doesn’t belong to the other set.
    - This means that one of those sets has an element, contradicting the definition of empty set (not having any elements). Q.E.D.. toInject asdfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff.
* Power Sets
  + Def: The power set of a set A is the set of all sugbsets of A
    - That is, the power set of A is {X | X ⊆ A}
  + Notation for the power set of A
    - P(A)
    - 2^A, we’ll see why a little later
  + 
* Subsets and Cardinality
  + If E ⊆ F, then |E| <|F|
  + 

1/18/2024

## Set Theory Foundations part II

Intersection of Sets (Binary Operation on sets *takes 2 sets and returns a set*)

* Let A and B be two arbitrary sets
* Def: The intersection (∩) of the sets A and B, denoted A ∩ B, is a new set made up of all the elements that belon to both A and B
  + A ∩ B = {x | x ∈ A and x ∈ B}
  + That is, A ∩ B is the overlap between A ∩ B
  + Ex:



* + - Let A = {1,2,3,a,b,c}, B = {1,3,a,d}
    - A ∩ B = {1,3,a}
* {𝑥|𝑥 is a 3-letter word} ∩ {𝑥𝑥|𝑥𝑥 is a word that starts with ‘a’} = {𝑥|𝑥 is a 3-letter word that starts with ‘a’} ={art, ask, and, ate, …}
* ℕ ∩ ℤ = ℕ
* If A and B share no elements, then 𝐴 ∩ 𝐵 = ∅
  + Such sets are called disjoint (or non-overlapping)

Union of Sets (idempotent)

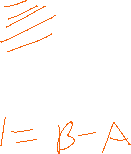
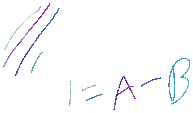
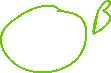
* Let A and B be two arbitrary sets
* Def: The Union (∪) of the sets A and B, denoted A ∪ B, is a new set made up of all the elements that belong to A or B or Both



* + A ∪ B = {x | x ∈ A or x ∈ B}
  + Example:
    - Let A = {1,2,3,a,b,c}, B = {1,3,a,d}
    - A ∪ B = {1,2,3,a,b,c ,d}

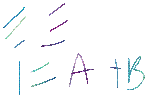
Difference of Sets

* Let A and B be two arbitrary sets
* Def: the difference (-) of the sets A and B, denoted A – B, is the set of all elements that belong to A but not to B



* + A – B = {x | x ∈ A and x ∉ B}
  + That is, A – B is A minus the over lap.

(symmetric) Sum of Sets



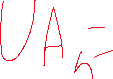
* A + B = (A-B) ∪ (B-A)
* A + B = (A∪B) – (A∩B)
* {x | (x ∈ A or x ∈ B) and x ∈ A∩B}

Universal Set

* Def: The universal set in a givern context is the set of all elements under dscussion
* Examples:
  + If a given lecture is focused entirely on integers, then the universal set is ℤ
  + Notation for Universal set: U
  + As a Ven diagram:
  + a Rectangle
  + Once a universal 𝑈 set is specified, then every set in that context is a subset of 𝑈
  + • For every set 𝐴 in that context, we have 𝐴 ⊆ U
* **Complement:** AC = Abar = {x ∈ U | x ∉ A } = U-A

Union and intersections of Indexed Colletions of Sets

* Given sets A1, A2, A3, … and n be a positive integer



* { x | x ∈ Ai for some i 1,2, … n }

Partitions

* Def: Sets A1, A2, A3, … are called mutually disjoint (or parwise disjoint) if and only if every two sets Ai and Aj are disjoint, i.e., Ai ∩ Aj = ∅
* Definition: Let 𝐴𝐴 be a set. A finite or infinite collection of sets 𝐴𝐴1, 𝐴𝐴2 , 𝐴𝐴3, … is called a partition of 𝐴 if both of the following conditions are met:
  + 𝐴 is the union of all the sets of the collection
  + A = ⋃𝑖 𝐴𝑖
  + The sets 𝐴1, 𝐴2 , 𝐴3, … are mutually disjoint
* Exercise:

A math exercise with numbers and text

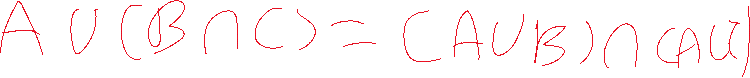
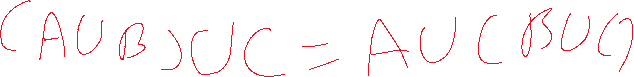
Description automatically generated with medium confidence

Properties of Set Operations

* Let A, B, and C be 3 arbitrary sets, U universal set
* Communtativity Laws: (∪ and ∩ are commutative)



* Associativity Laws: (∪ and ∩ are associative)



* Distributivity Laws:

U is distributive over ∩

* Identity Laws: 𝐴 ∪ ∅ = 𝐴, 𝐴 ∩ 𝑈 = 𝐴
* Complement Laws: 𝐴 ∪ 𝐴𝑐 = 𝑈, 𝐴 ∩ 𝐴𝑐 = ∅

A diagram of a mathematical equation

Description automatically generated with medium confidence

Pairs, Triplets, N-Tuples

* Let a, b, c, x1, x2, …, xn be a bunch of elements, not necessarly distinct
* Definitions
  + Couple / pair: (a,b)
  + Triplet / Triple: (a, b, c)
  + N-tuple: (x1,x2, …, Xn)
    - Xi is called the ith component of (x1, x2, …, xn)
  + Order matters

A close-up of a product set

Description automatically generated

1/23/2024

Why logic

* To learn to prove claims/statements rigorously
* To be able to judge better the soundness and consistency of (others’) arguments
* To learn to express certain statements. Semantics formally, precisely, concisely, and in a machine-readable form
  + For AI reasoning

# Propositions

* Any declarative statement (sentence) about which you can say it’s true or it’s false
  + Examples:
    - “Mars is a planet” is a proposition that is true
    - “Jupiter is a star” is a proposition that is false
* Questions, commands and exclamations are examples of sentences that are not proposititons
* **Truth Values:** They are values "t”ue" or T and “false (or F)
* Def: A simple proposition is proposition that conveys a single fact
* Def: A compound proposition is a proposition derived by negating and/or combining simple propositions using so-called logical connectives

# Logical Connectives

* Logical connectives are the logical operations “and”, “or”, and “not”
* “and” and “or” are binary operations, that is,
  + They take two operands where each operand s aproposition,
  + They result in a new proposition
* The “not” is a unary operation
  + It takes a single operand, which is a proposition
  + It results in a new proposition
* Examples of compound statements derived with logical compound statements derived with logical connectives



* + Mars is a planet and Jupiter s a star
  + Mars is a planet or Jupiter is a star
  + Not (Jupiter is a star)
    - In plain English, this statement means “Jupiter is not a star.”
* Mars is a planet and (Jupiter is a star or not(Pluto is a comet))
* Notations for logical Connectives
  + The operation “and” is denoted ∧
  + In Boolean algebra, it is denoted by the dot “•”
  + The operation “not” is denoted ¬, ~, (overbar), or ′ (prime)
  + Examples:
    - Mars is a planet and Jupiter is a star = Mars is a planet ∧ Jupiter is a star
    - Mars is a planet or Jupiter is a star = Mars is a planet ∨ Jupiter is a star
    - Mars is a planet and (Jupiter is a star or not(Pluto is a comet)) =
    - Mars is a planet ∧ (Jupiter is a star ∨ ¬(Pluto is a comet))

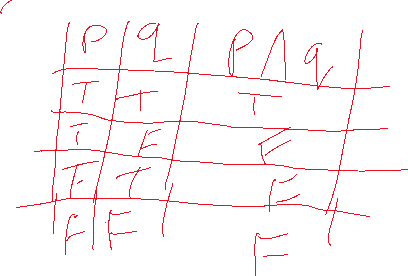
# Logic’s emphasis on form rather than substance

* Logic is concerned more about the truth values of proposiitno than about their meaning
* Therefore
  + Propositions are often denoted with sigle-letter symbols such as p, q, r, etc (convenience and efficiency)
  + The focus is on whether a propostion p is true or false, rather than on what p means/ desgnates
* For Example we can denote
  + “mars is a planet” by p
  + “Jupiter is a star” by q
  + “pluto is a comet” by r
* Then the above examples of compound statements ca be succinctly represented as :
  + Mars is a planet and Jupiter is a star”: 𝑝 ∧ 𝑞. 𝑝 ∧ 𝑞 is called a conjunction
  + “Mars is a planet or Jupiter is a star”: 𝑝 ∨ 𝑞. 𝑝 ∨ 𝑞 is called a disjunction
  + “not (Jupiter is a star)”: ¬q, ~q, 𝑞(overbar), or 𝑞𝑞′. called negations / complements
  + “Mars is a planet and (Jupiter is a star or not(Pluto is a comet))”: 𝑝 ∧ (𝑞 ∨ ¬𝑟).

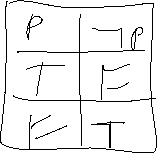
# Truth Tables

For the basic three logic operations

* The most basic truth tables, which form our foundation of logic and reasoning, are the truth tables for the three logical connectives:



|  |  |  |
| --- | --- | --- |
| p | q | p ∨ 𝑞 |
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |



Examples of Truth Tables ½

Truth table for 𝑝 ∧ (𝑞 ∨ 𝑟):

* The table will list the 8 different combinations of the truth values of p, q, and r.. toInject asdfffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff.
* Because it needs the (𝑞 ∨ 𝑟): values, a separate column (for (𝑞 ∨ 𝑟)) will be added to the table
* Finally, to fill the column of 𝑝 ∧ (𝑞 ∨ 𝑟), we apply the ∧ operation to the column of p and column of (𝑞 ∨ 𝑟):

A table of letters and numbers

Description automatically generated

Example of truth tables 2

A screenshot of a computer

Description automatically generated

Procedure for determining the equivalence of two propositions

1. Build a truth table for each propositions
2. Observe if the two columns of the two propositions are identical (make sure that the cominations of the truth-values are listed in the same order in both tables)
3. If identical, then the two propositions are equivalent; else they are not equivalent

A screenshot of a math table

Description automatically generated

• Since the columns of ¬(𝑝 ∧ 𝑞) and ¬𝑝 ∨ ¬𝑞 are identical, we conclude that ¬ 𝑝 ∧ 𝑞𝑞 and ¬𝑝 ∨ ¬𝑞 are equivalent

A white board with writing on it

Description automatically generated

Predicates and quantifiers

* Predicates generalize propositions to include quantification, that is, the phrases “for every” and “there exists”, and thereby increase the expressive and reasoning power of logic
* Universal quantifier:
  + It is the phrase “for every” (or “for all”)
  + Notation: ∀
* Existential quantifier:
  + It is the phrase “there exists” or “there is” (at least)
  + Notation: ∃

A black text on a white background

Description automatically generated

A close-up of a math test

Description automatically generated

A screenshot of a computer program

Description automatically generated

A blue paper with red and black text

Description automatically generated