Discrete Math Notes

Propositions, Negations, Conjunctions and Disjunctions

A **proposition** is a declarative statement that is either true or false

- Eg, "The sky is blue", " The moon is made of cheese"
- Example of what is *not* a proposition, "Sit down", "X+1 = 2". The last one is not a proposition because it is not true *or* false

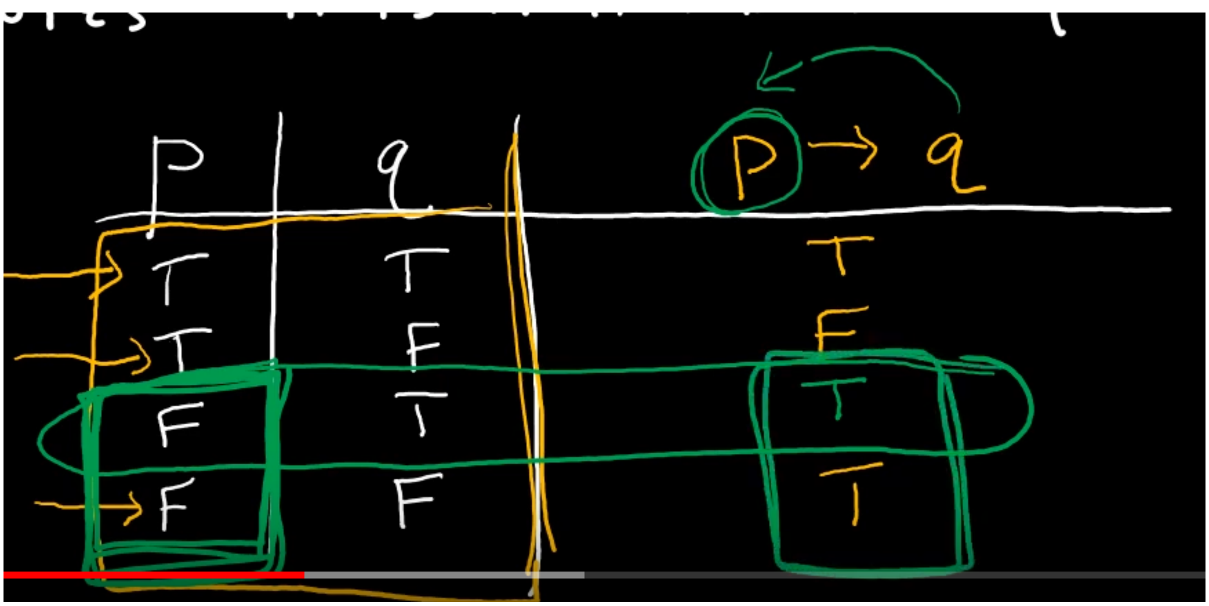
A compound propoisition is comprised of propositions and one or more of the following connectives

- Negation ¬ "NOT"
- Conjuction ^ "AND"
- Disjunction v "or"
- Implication → "If, then"
- Bicondition ↔ "if and only if"

Implications (Converse, inverse, contrapositive) and Biconditionals

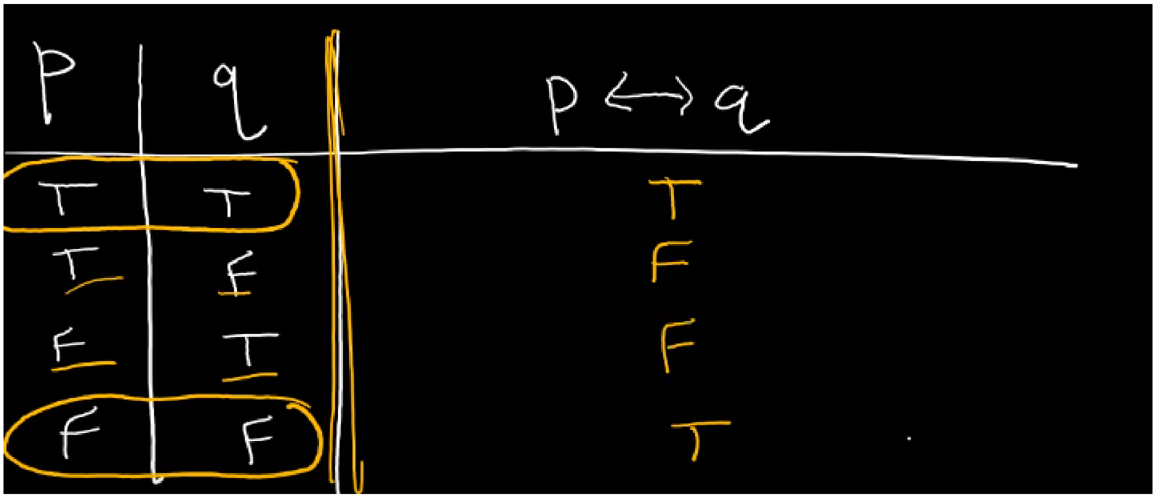
Implication (conditional statement)

- When the hypothesis is true, the conclusion must be true for the implication to be true.



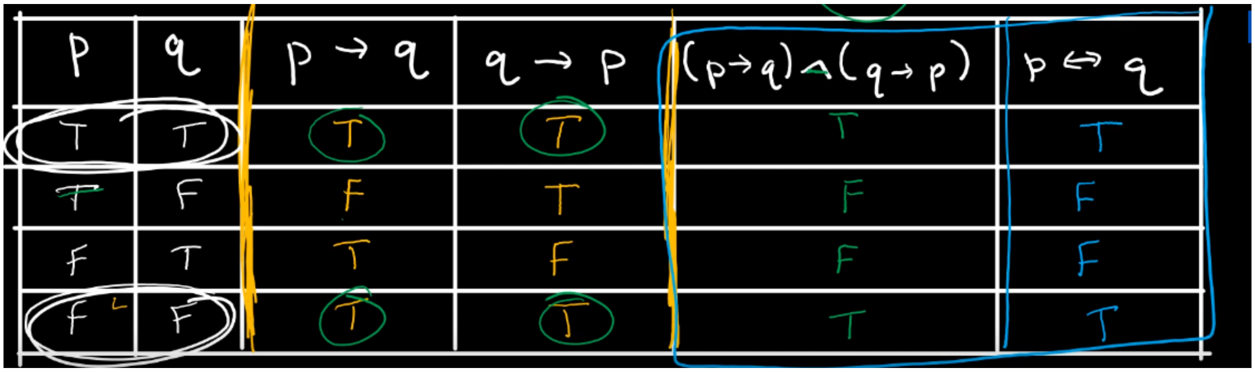
- Converse, if the implication is p → q, than the converse is q → p (switching order)
- Inverse, if the implication is p → q, than the inverse is ¬p → ¬q (negate)
- Contrapositive, if the implication is p → q, then the contrapositive is ¬q → ¬p (switching the order and negate). Contrapositive will always have the same truth values as the implication

For a biconditional to be true, both propositions must share the same truth value



p	q	p ↔ q
T	T	T
T	F	F
F	T	F
F	F	T

Overall



p	q	p → q	q → p	(p→q)∧(q→p)	p ⇔ q
T	T	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	F
F	F	T	T	T	T

1.1.3 Constructing a truth table for compound propositions

You need 2^{number of propositions} rows

You need a column for each propositional value, a column for the truth value of each expression that occurs in the compound proposition as it's built up, and lastly a column for the final result.

The order of precedence (importantce/priority) for operators is as following:

- ¬
- ^

- v
- →
- ↔

Example

The proposition is p v q → ¬r

1) first constructi columns for each propositions: p, q, r

2) create a colum for each compound propisition: p v q, ¬r

3) create a column for the final compound proposition

4) create 2^3 = 8 rows

1.2.1 Translating propositional logical statements

equate each condition to a letter

the letter must always represent the **positive** outcome

Eg. " You can get a free sandwich on thursday if you buy a sandwich or a cup of soup":

- P: I buy a sandwhich
- Q: I buy a cup of soup
- R: I get a free sandwhich on thursday

Answer: (P v Q) → ¬R

1.2.2 Solving Logic Puzzles

Context:

- Knights always tell the truth
- Knaves always lie

Puzzle:

- A says "B is a knight"
- B says "The two of us are opposite types"
- What are A and B?

Answer:

- P: A is a knight
- q: B is a knight
- 4 Possible outcomes
  - p ^ q
  - p ^ ¬q
  - ¬p ^ q

- ¬p ^ ¬q

Go through each one, the only one that doesnt contradict is the last one

1.2.3 Introduction to Logic circuits

- NOT gate
- OR gate
- AND gate

A Logic circuit is designed to perform complex tasks designed in terms of elementary logic functions.

1.3.1 "Proving" logical Equivalences with Truth tables

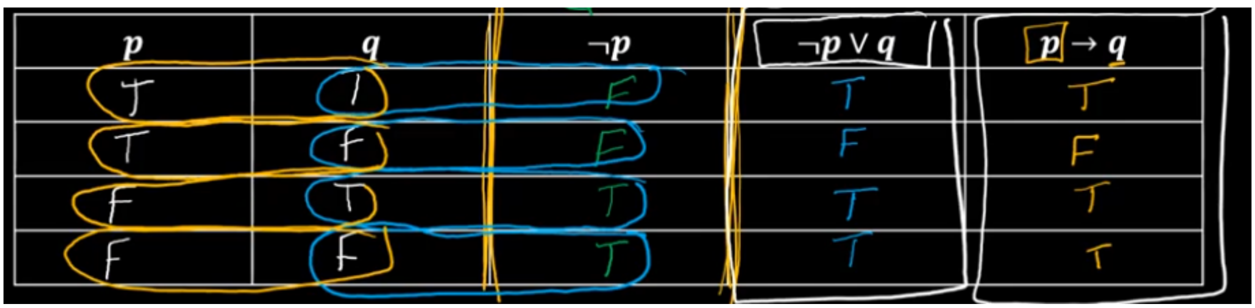
a **tautology** is a proposition which is always true. Eg `p v ¬p`

a **contradiction** is a proposition which is always false. Eg `p ^ ¬p`

a **contingency** is a proposition which is neither a tautology nor a contradiction. Example: `p`

Logical equivalences

- Two compound proposition, p and q, are logically equivalent if p ↔ q is a tautology
- Eg `¬p v q === p -> q` which can be proven with a truth table



p	q	$\neg p$	$\neg p \vee q$	$p \rightarrow q$
T	T	F	T	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

1.3.2 Key Logical Equivalences Including De Morgan's Laws

Identity Laws — T = Tautology (True), F = (contradiction) False

- P ^ T === P
- P v F === P

Domination Laws

- P v T === T
- P ^ F === F

Indempotent Laws

- P v P === P
- P ^ P === P

Double Negation Law

- ¬(¬P) === P

Absorbtion Laws

- P v (P^Q) === P
- P ^ (PvQ) === P

Negation Laws

- P v ¬P === T
- P ^ ¬P === F

Commutative Laws

- P v Q === Q v P
- P ^ Q === Q ^ P

Distributive laws

- P v (Q ^ R) === (P v Q) ^ (P v R)
- P ^ (Q v R) === (P ^ Q) v (P ^ R)

Associative Laws

- (P v Q) v R === P v (Q v R)
- (P ^ Q) ^ R === P ^ (Q ^ R)

De Moran's Laws

- ¬(P v Q) === ¬P ^ ¬Q
- ¬(P v Q) === ¬P ^ ¬Q