An argument is a sequence of sentences, the last member of which is the conclusion and the earlier members of which are the premises.

$$<\underbrace{s_1,....s_n}_{\text{premises}}, \overbrace{s}^{\text{conclusion}}>$$

Some examples of arguments

Argument 1

1. Jill and Bill both have jobs.
2. If Jill has a job, then she commutes.

C. Either Jill commutes or she doesn't own a car.

Argument 2

1. Jill and Bill both have jobs.

C. Jill commutes.

More arguments:

(1) If God exists, then there is an omnipotent, benevolent being.

(2) If there is an omnipotent, benevolent being, then there is no suffering

So, (C) God exists only if there is no suffering

1. Kant is a transcendental idealist, unless Hume is an empiricists.
2. Either Leibniz is not a rationalist or Hume is an empiricists.
3. Leibniz is a rationalist.

So, (C) Kant is a transcendental idealist.

An argument is **valid** just in case it is absolutely impossible for all of its premises to be true while its conclusion is false.

A function is a relation from inputs to outputs. So, for example, the addition function in arithmetic takes a pair of numbers as inputs and delivers their sum as an output.
A truth-functional operator takes truth-values as inputs and delivers a truth-value as output. It is a function, then, on the truth-values on the simpler components of a sentence used to determine the complex sentence's truth-value.

Consider the conclusion of argument 1: Either Jill commutes or she doesn't own a car. This sentence is "built" from two simpler sentences:

Jill commutes

Jill owns a car

The expressions 'either…or' and 'not' (which is contradicted in the helping verb 'does') express truth-functions and are truth-functional operators.

Negation is the simplest truth-function. It is a "truth-value flipper." So, if 'not' is attached to a true sentence, then the resulting complex negated sentence is false; if 'not' is attached to a false sentence, then the resulting complex negated sentence is true.

Two assumptions: 1. There are no "truth-value gaps." That is, absolutely every sentence is either true or false.

(Some think that vagueness provides a counterexample. So, suppose that Bill is a borderline case of being bald: He isn't clearly bald but he also isn’t clearly not bald. Then ‘Bill is bald’, some say, is neither true nor false.)

2. Noncontradiction: No sentence is both true and false.

(Some think that certain paradoxes involving the truth predicate and set theory are best resolved by accepting true contradiction; this is to accept a *paraconsistent logic*. For example, some say that the liar sentence ‘This sentence is not true’ is a “true contradiction.”)

Back to negation: In English, negations typically attach (at least in their surface forms) to verbs. So, ‘doesn’t own a car’ is short for ‘does not own a car’ and the negation appears to operate on the verb-phrase ‘owns a car’. However, verb-phrases don’t have truth-values; only complete sentences have truth-values. So, for our purposes, we will recast negations as “really” operating on the entire sentence. So,

Jill doesn’t own a car

will be seen as

$$\underbrace{\text{It is not the case that}}_{\text{negation operator}}\ \overbrace{\text{Jill owns a car}}^{\text{simpler input sentence}}$$

Then the negated sentence is true exactly when the simpler input sentence is false.

In general, where *s* is any sentence whatsoever,
'It is not the case that *s*' is true exactly when *s* is false.

'not' is a **one-place** truth-function: Grammatically, it takes a single sentence as input and delivers a complex negated sentence as output; semantically, it takes a single truth-value as input and delivers a truth-value as output.

'Either….or' is a **two-place** truth-function.
Grammatically, it combines a pair of sentences to create a more complex **disjunction**.

Inputs: Jill commutes
Jill doesn't own a car [read: It is not the case that Jill owns a car]

Output: The disjunction
Either Jill commutes or it is not the case that Jill owns a car.

The two inputs are called the disjunction's **disjuncts**. So, our output's right disjunct is 'It is not the case that Jill owns a car' and its left disjunct is 'Jill commutes'.
A disjunction is true exactly when at least one of its disjuncts are true.

We will use five truth-functional operators in our formal language, *the language of propositional logic*. They are the following:

type	canonical English form	symbol
negation	not	$\neg$
conjunction	and	$\wedge$
disjunction	or	$\vee$
conditional	if, then	$\rightarrow$
bi-conditional	if and only if	$\leftrightarrow$

These five words (and their cognates) are our "logic words." In translating English sentences into our formal language, the first step is to **identify the logic words**.

Argument 1

1. Jill and Bill both have jobs.
2. If Jill has a job, then she commutes.

C. Either Jill commutes or she doesn't own a car.

(Some "logic words," like 'if' and 'then', really belong together and are a "single" logic word…. although 'if' can occur by itself and mean exactly the same.)

The second step is to **identify the atomic** sentences from which the sentences are composed. The atomic sentences are the shortest units that do not contain logic words and yet are sentences (and so have truth-values).

1. Jill and Bill both have jobs.

    Atomic parts:

    Jill has a job.

    Bill has a job.

2. If Jill has a job, then she commutes.

    Atomic parts:

    Jill has a job.

    Jill commutes.

C. Either Jill commutes or she doesn't own a car.

    Atomic parts:

    Jill commutes.

    Jill owns a car.

The third step is to **map atomic parts onto atomic sentence letters from the language of Propositional Logic**. When a given sentence occurs multiple times in a sentence or throughout an argument it is crucial that *all* *occurrences are given the same atomic sentence letter*, otherwise the translation will be inadequate.

In our example, there are four atomic sentences:

Jill has a job.	p
Bill has a job.	q
Jill commutes.	r
Jill owns a car.	S

The fourth step is to **replace the atomic sentences in the target sentences with their associated atomic sentence letters**, leaving the logic words in their English form.

1. Jill and Bill both have jobs.
   p and q
2. If Jill has a job, then she commutes.
   If q, then r

C. Either Jill commutes or she doesn't own a car.
   Either r or not s

The final step is to **translate the logic words** and **decide their relative order (or scope).**

1. Jill and Bill both have jobs.

   p and q

   $(p \wedge q)$

2. If Jill has a job, then she commutes.

   If q, then r

   $(q \rightarrow r)$

C. Either Jill commutes or she doesn't own a car.

   Either r or not s

   $(r \vee \neg s)$