HANDOUT #2 – PROPOSITIONAL LOGIC –SYMBOLS, SYNTAX, SEMANTICS, AND TRANSLATION

The language of propositional logic (hereafter 'PL') consists of a set of **symbols**, a set of **formation rules** (a syntax) that tells us whether a formula in PL is well-formed (grammatically correct), and a **semantics** that assigns formulas a truth value.

PL: Symbols

PL consists of the following symbols:

(1) an infinite number of upper-case Roman (unbolded) letters with or without subscripted integers (e.g. A_1, A_2, A_3, B, C, …, Z)
(2) five truth-functional operators ($\vee$, $\rightarrow$, $\leftrightarrow$, $\neg$, $\wedge$)
(3) parentheses, braces, and brackets to indicate the scope of truth-functional operators

PL: Syntax

An *atomic proposition* is a well-formed formula in PL consisting only of a single propositional letter, e.g. P, R, S, D. A *complex proposition* is a well-formed formula in PL that contains at least one propositional letter and a truth-functional operator. A *well-formed formula* (abbreviated as 'wff', pronounced 'woof') in PL is a proposition that is capable of being generated by one of the seven formation rules below.

1 Every propositional letter (e.g. P, Q, R) is a wff.
2 If **P** is a wff, then $\neg$**P** is a wff.
3 If **P** and **R** are wffs, then (**P** $\wedge$ **R**) is a wff.
4 If **P** and **R** are wffs, then (**P** $\vee$ **R**) is a wff.
5 If **P** and **R** are wffs, then (**P** $\rightarrow$ **R**) is a wff.
6 If **P** and **R** are wffs, then (**P** $\leftrightarrow$ **R**) is a wff.
7 Nothing else is a wff except what can be formed by repeated applications of 1–6.

To illustrate how the above formation rules can be used to create complex propositions, consider the following use of formation rules to show that '$\neg P \rightarrow R$' is a wff:

1	Every propositional letter is a wff, so P and R are wffs.	Rule 1
2	If P is a wff, then $\neg$P is a wff.	Line 1 + Rule 2
3	If $\neg$P and R are wffs, then $\neg P \rightarrow R$ is a wff.	Lines 1,2 + Rule 5

Each of the formation rules results in the creation of a particular type of wff.

1 Propositions P, Q, R are atomic propositions
2 A proposition $\neg$**P** is a negated proposition
3 A proposition **P**$\wedge$**R** is a conjunction
4 A proposition **P**$\vee$**R** is a disjunction
5 A proposition **P**$\rightarrow$**R** is a conditional
6 A proposition **P**$\leftrightarrow$**R** is a biconditional

Classroom Exercises
1. ¬P
2. P∧¬R
3. ¬(P→¬R)

Scope Indicators & the Main Operator

Truth-functional operators of PL have scope. That is, they operate over a specified range of propositions. The negation operator '¬' operates over the proposition (atomic or complex) to its immediate right while all of the other truth-functional operators in PL operate over the propositions to their immediate left and right. For example, the negation sign in '¬P' operates on 'P' while the carrot in 'P∧R' operates on both 'P' and 'R'. Scope indicators like parentheses are used so that the scope of operators is unambiguous. To see why this is necessary consider the following wff:

(1) ¬P→R

It is unclear whether the sign for negation is supposed to operate upon 'P' or the complex proposition 'P→R'. Using scope indicators, we can ensure that the scope of the sign for negation is unambiguous:

(1) ¬P→R
(2) ¬(P→R)

In (1), the sign for negation applies only to 'P' whereas in (2) it applies to the complex proposition 'P→R'.

The same is true for all other truth-functional operators. Consider (3) below:

(3) P∧R∨S

It is ambiguous whether the carrot '∧' operates on 'P' and 'R∨S' or operates on just 'P' and 'R'. Likewise, it is ambiguous whether the wedge '∨' operates on 'S' and 'P∧R' or just 'S' and 'R'. The use of scope indicators allows for making the scope of these operates unambiguous:

(4) P∧(R∨S)
(5) (P∧R)∨S

The **main operator** of a PL formula is the truth-functional operator with the greatest scope. The formation rules can be used to determine the main operator of a wff. To see this clearly, first notice that other than Rule 1, all of the formation rules are associated with a truth-functional operator, e.g. Rule 6 with '↔'. Second, the main operator of a wff is the truth-functional operator associated with the *last* formation rule applied to create the wff. For example, consider the use of the formation rules to show that '¬(P→¬R)' is a wff:

1	Every propositional letter is a wff, so P and R are wffs.	Rule 1
2	If R is a wff, then $\neg$R is a wff.	Line 1 + Rule 2
3	If P and $\neg$R are wffs, then P$\rightarrow\neg$R is a wff.	Lines 1, 2 + Rule 5
4	If P$\rightarrow\neg$R is a wff, then $\neg$(P$\rightarrow\neg$R) is a wff.	Line 3 + Rule 2

Notice that the last formation rule applied is Rule 2 (associated with '$\neg$'). The '$\neg$' applied to 'P$\rightarrow\neg$R' is the main operator.

Literal Negation

The literal negation of **P** is a proposition of the form $\neg$(**P**). It is the proposition that results from applying formation rule (2) on a proposition. It is important to note that the literal negation of a proposition **P** is the negation of the *entire formula* and not just *part of the formula*.

	Proposition **P**	Literal negation of **P**
1	P	$\neg$P
2	P$\rightarrow$R	$\neg$(P$\rightarrow$R)
3	$\neg$P$\wedge$R	$\neg$($\neg$P$\wedge$R)

Classroom Exercises

A. Determine the main operator of the following propositions:

1. $\neg$P
2. $\neg\neg$P$\wedge\neg$R
3. $\neg$($\neg$P$\leftrightarrow\neg$R)

B. Determine the literal negation of the following propositions:

1. $\neg$P
2. $\neg$P$\wedge\neg$R
3. $\neg$($\neg$P$\rightarrow\neg$R)

PL: Semantics

Well-formed formula can be assigned one (and only one) *truth value*: true (T) or false (F). This is known as the **principle of bivalence**. We call the assignment of a truth value to a wff a **valuation** or **truth-value assignment**.

The truth values of atomic propositions are stipulated, i.e., we simply assign propositional letters truth values. For example,

v(P)=T

The above states that 'P' is valuated as *true*. In other words, 'P' is given 'T' (or True) as its truth-value assignment. Alternatively, to indicate that a proposition is *false*, we write the following:

v(R)=F

The above states that ‘R’ is valuated as *false*. In other words, ‘R’ is given ‘F’ (or False) as its truth-value assignment.

Determining the truth values of complex propositions (propositions involving at least one truth-functional operator) is determined entirely by the truth values of the propositional letters that compose the propositions and truth functions associated with the truth-functional operators. In other words, PL is a *truth-functional language*. What this means is that the truth value of complex propositions (propositions with at least one truth-functional operator) are determined entirely by the propositions that compose it. Just as the numerical output (the sum) of arithmetical equation is determined by the numerical input (the numbers), the truth-value output of complex wff is determined by the truth-value input (the truth values of the propositions that compose it).

Negation

In the case of negated propositions **¬P**: **¬P** is true if and only if **P** is false; **¬P** is false if and only if **P** is true.

P	**¬P**
T	F
F	T

Conjunctions

In the case of conjunctions **P∧R**: **P∧R** is true if and only if **P** is true and **R** is true; **P∧R** is false in all other cases.

P	**R**	**P∧R**
T	T	T
T	F	F
F	T	F
F	F	F

Disjunctions

In the case of disjunctions **P∨R**: **P∨R** is false if and only if **P** is false and **R** is false; **P∨R** is true in all other cases.

P	**R**	**P∨R**
T	T	T
T	F	T
F	T	T
F	F	F

Conditionals

P	**R**	**P→R**
T	T	T
T	F	F
F	T	T
F	F	T

Biconditionals

P	**R**	**P↔R**
T	T	T
T	F	F
F	T	F
F	F	T

PL: Translation

We have at our disposal two languages: an abstract truth-functional language (**PL**) and a natural language (**English**). What we focus on here is how these two languages relate to each other.

Single propositional letters (atomic propositions) in PL can be used to capture simple English sentences that express propositions that do not contain the use of truth-functional uses of expressions like ‘not’, ‘and’, ‘or’, etc. (this will become a little clear as you consider the remaining truth-functional operators).

John is kind.	J
John will go to heaven	H
John went to the store.	S

Negated propositions **¬P** in PL capture English uses of ‘not’, ‘it is not the case that’, ‘it is false that’.

John is *not* kind.	¬J
It is not the case that John will go to heaven	¬H
John *did not* go to the store.	¬S

For consider that a proposition *John is not kind* is true just in the case that *John is kind* is false and *John is not kind* is false just in the case that *John is kind* is true.

Conjunctions **P∧Q** in PL capture truth-functional uses of ‘and’ in English.

John is kind *and* John will go to heaven.	J∧H
John is kind *and* John will go to the store.	J∧S
John is kind, John will go to heaven, and John went to the store.	(J∧H)∧S

For consider that a proposition *John is kind and John will go to heaven* is **true** just in the case that *John is kind* is **true** and *John will go to heaven* is **true**. It is **false** if either of those propositions (or both) are **false**.

Disjunctions **P∨Q** in PL capture the inclusive truth-functional use of 'or' in English:

John is kind *or* John will go to heaven.	J∨H
John is kind *or* John went to the store.	J∨S
John is not kind *or* John will not go to heaven.	¬J∨¬H

For consider that a proposition *John is kind or John will go to heaven* is **false** if and only if *John is kind* is **false** or *John will go to heaven* is **false**. In all other cases, this proposition is true. That is:

> If *John is kind* is **true**, then *John is kind or John will go to heaven* is **true**.
> If *John will go to heaven* is **true**, then *John is kind or John will go to heaven* is **true**.
> If *John is kind* is **true** AND *John will go to heaven* is **true**, then *John is kind or John will go to heaven* is **true**.

Conditionals **P→Q** in PL capture the truth-functional use of 'if **P**, then **Q**' propositions in English:

If John is kind, then John will go to heaven.	J→H
If John is kind, then John went to the store.	J→S
If John is in Toronto, then John is in Canada	J→C

For consider that a proposition *if John is kind, then John will go to heaven* is **false** if and only if *John is kind* is **true** and *John will go to heaven* is **false**. In all other cases, this proposition is true.

> If *John is kind* is **true** and *John goes to heaven* is **true**, then *if John is kind, then John will go to heaven* is **true**.
> If *John is kind* is **false** and *John goes to heaven* is **true**, then *if John is kind, then John will go to heaven* is **true**.
> If *John is kind* is **false** and *John goes to heaven* is **false**, then *if John is kind, then John will go to heaven* is **true**.

It is sometimes unclear why truth-functional uses of 'if **P**, then **Q**' is true, when **P** is false. But suppose that God tells John that *if John is kind, then John will go to heaven*. We want to know under what conditions God keeps his promise.

Suppose that John is not kind. John makes life harder on his neighbors, steals from the poor, and scowls at everyone he sees. But, when John dies, God sees what a pitiful wretch he is and decides to send him to heaven anyway. Has God, in sending John to heaven, *broken* his promise? That is, has God said something **false** in uttering *if John is kind, then John will go to heaven*? The answer is no. God never claimed that he would send John to heaven ***if and only if*** he was kind. Rather, God's promise is kind of a guarantee that if John is kind, he will send him to heaven, but even if he isn't, God might send him anyway. A conditional is like this: for if 'P' is false, but 'R' is true, then 'P→R' is true.

P	**R**	**P→R**
T	T	T
T	F	F
F	T	T
F	F	

Finally, suppose that John is not kind. Again, John is a cruel person, diabolical to the core. And, when John dies, God sees what a pitiful wretch John is and decides *not* to send him to heaven. Instead, God throws him into the darkest corners of hell. Has God, in sending John to hell, broken his promise? The answer is no. God's promise to John was that if John is kind, then God will send John to heaven, but John wasn't kind, so God is under no obligation to send John to heaven. A conditional is like this. If 'P' is false and 'R' is false, then 'P→R' is true.

P	**R**	**P→R**
T	T	T
T	F	F
F	T	T
F	F	T

Biconditionals **P↔Q** in PL capture the truth-functional use of '**P** if and only if **Q**' propositions in English:

John is kind *if and only if* John will go to heaven.	J↔H
John is kind *if and only if* John went to the store.	J↔S
John is in Toronto *if and only if* John is in Canada	J↔C

Propositional Logic Symbols, Syntax, Semantics: A Brief Overview

Atomic Propositions

Atomic Proposition	P, Q, R (any upper-case Roman letter without a truth-functional operator)

Syntax
Any propositional letter ‘P’, ‘Q’, ‘R’, …, ‘Z’ is a well-formed formula.

Semantics

P	**P**
T	T
F	F

A proposition ‘P’ is true if and only if P.

Translation
Atomic propositions in PL capture simple English sentences that express propositions that do not contain the use of truth-functional uses of expressions like ‘not’, ‘and’, ‘or’, etc.

John is happy.	J
It is the case that Mary is well.	M
John went to the store.	S

Negated Propositions

Negated Proposition	**¬P**
‘¬’ sign of negation	

Syntax
The negation ‘¬’ operates on the proposition (atomic or complex) to its immediate right. For example, the negation in ‘¬P∧R’ applies to the atomic ‘P’ but the negation in ‘¬(P∧R) applies to the complex ‘P∧R’.

Semantics

P	**¬P**
T	F
F	T

A negated proposition **¬P** is true if and only if the proposition it operates upon **P** is false, and a negated proposition **¬P** is false if and only if the proposition it operates upon **P** is true.

Translation
Negated propositions in PL capture English uses of ‘not’, ‘it is not the case that’, ‘it is false that’.

John is not happy.	¬J
It is not the case that Mary is well.	¬M
John didn't go to the store.	¬S

Conjunctions

Conjunction	**P∧R**
'∧' = the carrot, sign for conjunction	
P is the *left conjunct*	
R is the *right conjunct*	

Syntax
The carrot '∧' operates on the propositions (atomic or complex) to its immediate left and right. For example, the carrot in 'P∧R' applies to the atomic 'P' and the atomic 'R'. In '(P→R)∧¬P, the carrot applies to 'P→R' and '¬P'.

Semantics

P	**R**	**P∧R**
T	T	T
T	F	F
F	T	F
F	F	F

A conjunction is true just in one case: when both of the conjuncts **P** and **R** are true.

Translation
We can use conjunctions in PL to capture English uses of 'and'.

John is happy and Mary is well.	J∧M
Mary is well and John is happy.	M∧J
John is not happy and Mary is not well.	¬J∧¬M

Disjunctions

Disjunction	**P∨R**
'∨' = the wedge, sign for disjunction	
P is the *left disjunct*	
R is the *right disjunct*	

Syntax
The wedge '∨' operates just like the carrot '∧', i.e. on propositions (atomic or complex) to its immediate left and right. For example, the wedge in 'P∨R' applies to the atomic 'P' and the atomic 'R'. In '(P→R)∨¬P, the wedge applies to 'P→R' and '¬P'.

Semantics

P	**R**	**P∨R**
T	T	T
T	F	T
F	T	T
F	F	F

A disjunction is false just in one case: when both of the disjuncts **P** and **R** are false.

Translation
We can use disjunctions in PL to capture the inclusive use of ‘or’ in English:

John is happy or Mary is well.	J∨M
John will have cake or ice cream.	C∨I
Either John has rabies or he has the mumps	J∨M

Conditionals*

Conditional	**P→R**
‘→’ = the arrow, sign for conditional	
P is the *antecedent*	
R is the *consequent*	

Syntax
The arrow ‘→’ operates just like the carrot ‘∧’ and the wedge ‘∨’, i.e. on propositions (atomic or complex) to its immediate left and right. For example, the arrow in ‘P→R’ applies to the atomic ‘P’ and the atomic ‘R’. In ‘(P∧R)→ ¬P, the arrow applies to ‘P∧R’ and ‘¬P’.

Semantics

P	R	P→R
T	T	T
T	F	F
F	T	T
F	F	T

A conditional is false just in one case: when the antecedent **P** is true and the consequent **R** is false.

Translation
We can use conditionals in PL to capture a truth-functional use of ‘if **P**, then **R**’ in English:

If John is in Toronto, then John is in Canada	J→C
If Mary is a lawyer, then Mary makes a lot of money.	L→M
If John has rabies, then he does not have the mumps	R→¬M

Biconditionals

Biconditional	**P↔R**
'↔' = double arrow, sign for biconditional	
P is the *left-hand side*	
R is the *right-hand side*	

Syntax
The double arrow '↔' operates just like the carrot '∧', the wedge '∨', and the arrow '→', i.e. on propositions (atomic or complex) to its immediate left and right. For example, the double arrow in 'P↔R' applies to the atomic 'P' and the atomic 'R'. In '(P∧R)↔¬P, the double arrow applies to 'P∧R' and '¬P'.

Semantics

P	**R**	**P↔R**
T	T	T
T	F	F
F	T	F
F	F	T

A biconditional is true just in the case where the left-hand side **P** and the right-hand side **R** have the same truth value (be it true or false).

Translation
We can use biconditionals in PL to capture a truth-functional use of '**P** if and only if **R**' in English:

John is in Toronto if and only if John is in Canada	J↔C
Mary has a heart if and only if she has a kidney.	H↔K
Liz will go to the Penn State if and only if she has good grades.	L↔G

Classroom Exercises

Translate the following propositions into PL
1. John is a zombie.
2. John is a zombie and Mary is happy.
3. If John is a zombie, then Mary is happy.
4. If John is not a zombie, then Mary is not happy.
5. Either John is a zombie or Mary is happy.
6. John is a zombie if and only if Mary is happy.
7. If John is a zombie and Mary is happy, then either John is a zombie or Mary is not happy.
8. It is not the case that John is not a zombie.