# Logic & Arguments

Part 3

# Logic Statements

Make Mr. Spock Proud

## Logic Statements

- Logic is used to construct all proofs and computer systems
- A *statement* is any declarative sentence that results in either true or false



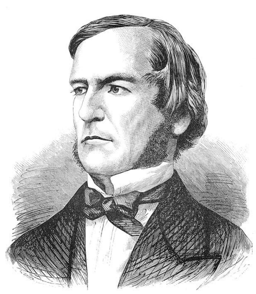
## Examples of Statements

- *There are exactly 35 people in this room*
- *Sacramento State is located next to a river*
- *10 + 2 = 11*
- *We had great choices for the 2016 Election*



## Boolean Logic

- Discovered by George Boole
- First published in *The Mathematical Analysis of Logic (1847)*
- Revolutionized logic & proofs and is part of framework of modern of computer science





## Boolean Operators

- Statements can be combined in *compound statements* using Boolean operators
- After statements are combined, they are still statements
- For example: "p and q"
  - given that p and q are both statements
  - then "p and q" is also a statement

## Let's Review Boolean Operators

| Operator | Name |
|---|---|
| `p and q` | True only if both p and q are true |
| `p or q` | True if either p or q true |
| `not p` | True if p false |
| `p xor q` | True if p and q are different |
| `p implies q` | True unless p is true and q is false |



## Logic Notation of Operators

| Code-like | Logic |
|---|---|
| `p and q` | $p \land q$ |
| `p or q` | $p \lor q$ |
| `not p` | $\neg p$ |
| `p xor q` | $p \oplus q$ |
| `p implies q` | $p \rightarrow q$ |



## Truth Table

| $p$ | $q$ | $\neg p$ | $p \lor q$ | $p \land q$ | $p \rightarrow q$ |
|---|---|---|---|---|---|
| T | T | F | T | T | T |
| T | F | F | T | F | F |
| F | T | T | T | F | T |
| F | F | T | F | F | T |



## AND Example

True True

$$1 < 2 \land 10 < 40$$

Result: True



## AND Example 2

True False

$$1 < 2 \land 12 < 10$$

Result: False



## OR Example

True True

$$1 < 2 \lor 10 < 40$$

Result: True

## OR Example 2

True · False

`5 > 3 ∨ 44 < 8`

Result: True



## OR Example 3

False · False

`10 < 2 ∨ 12 < 10`

Result: False



## NOT Example

True

`¬(1 < 2)`

Result: False



## NOT Example

False

`¬(1 = 2)`

Result: True



## Examples

| | |
|---|---|
| `1 < 3 ∧ 10 < 40` | True |
| `1 = 3 ∧ 10 < 40` | False |
| `¬ (12 ≠ 12)` | True |
| `1 > 3 ∨ 30 < 20` | False |



## Implication

What is the implication of a little arrow?

## Implication

- The only Boolean operator that causes confusion is implication
- However, its usage is ***vital*** to understand – since it is used your programs (even if you might not see it)



## Implication

- For "p implies q"…
- p is called the *antecedent* (or hypothesis or assumption)
- q is called the *consequent* (or conclusion)



## Implication

- "p implies q" is contradicted (false) **only** when…

  **p is true** and **q is false**

- In all other cases, it is true



## Implies Example

True — True

**1 < 2 → 10 < 40**

Result: True



## Implies Example 2

True — False

**5 > 3 → 44 < 8**

Result: False



## Implies Example 3

False — False

**10 < 2 → 12 < 10**

Result: True

## Implication

- Consider the expression below
- The word "then" is alternative way of saying "implies"
- So, Is it True? False?

**if x > 2 then $x^2$ > 4**

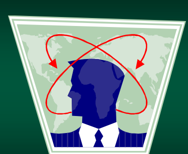


## Implication

- There <u>are</u> different values of x that will make the antecedent and consequent both true and false
- …but nothing that makes the expression false
- It is always true - no matter what the value of x

**if x > 2 then $x^2$ > 4**



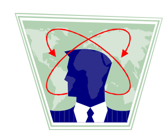


## Analyzing Implication

Understanding is Power

## Analyzing Implication

- Implication is both simple and complex
- It is used in <u>all</u> aspects of logical proof and the basis of all programs
- Understanding is complexity is essential to understanding logic (and discrete math)





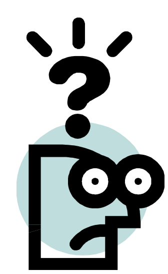
## Implication Examples

- *If the Moon is made of cheese then the Moon is a tasty snack.*
- *If the flag has a bear then it is the Flag of California.*
- *If it is a fish then it is lives in water.*
- *If the university is Sacramento State then the mascot is a hornet.*



## Many Ways to Say "Implies"

- *A implies B*
- *$A \rightarrow B$*
- *B if A*
- *If A then B*
- *B given A*



## "If I pan for gold then I'll get rich"

- Let's look at this statement closer
- It can be rewritten *"Pan of Gold→ Get Rich"* or, very tersely, *"P → R"*
- There four different combinations of the truth table



## True → True

- If P is true, and R is true…
- *"We panned for gold and got rich"*
- Statement is true
  - we asserted that if P is true then R is true
  - since both are true, the statement is affirmed
  - true → true = true



## False → True

- What if P is false and R is true
- *"We didn't pan for gold and got rich"*
- Statement is true
  - the fact we got rich (without panning for gold), doesn't mean that our statement is false
  - it has not contradicted the statement
  - false → true = true



## False → False

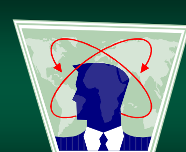
- What if P is false and R is false?
- *"We didn't pan for gold and didn't get rich"*
- Statement is true
  - the fact both are false, still does not contraction our original statement
  - it stated "IF we pan for gold THEN we get rich"
  - false → false = true



## True → False

- Finally what if P is true and R is false?
- *We panned for gold, but didn't get rich*
- Statement is false
  - we asserted if P is true then R must be true
  - however, since this contradicts the assertion, the result of the implication is false
  - true → false = false



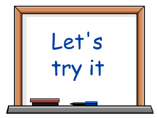


## Deconstructing Implication

Other operators; same logic

## Deconstructing Implication

- The implication logic can be broken down into the forms that are easier to remember
- This is actually quite important when we cover a few logical tricks later one
- So, let's look at the truth table for other logical operators






## Analyzing Implication

- So, can implication be written using just logical "and", "or", or "not"?
- Yes, we can!

$$p \rightarrow q$$



## Truth Table – One way to do it

| p | q | p → q |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |

True whenever p is false

True whenever q is true



## Analyzing Implication

- So, we can definite p → q as "not p or q"
- Like before, let's prove in our truth table

$$p \rightarrow q \equiv \neg p \lor q$$



## Truth Table – Or Logic

| p | q | ¬ p | ¬p ∨ q | p → q |
|---|---|---|---|---|
| T | T | F | T | T |
| T | F | F | F | F |
| F | T | T | T | T |
| F | F | T | T | T |



## Truth Table – One way to look at it

| p | q | p → q |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |

Only false for:

$p \land \neg q$

We can negate this.

## Analyzing Implication

- So, we can definite p → q as "not (p and not q)"
- It doesn't look quite right, let's test it out

$$p \rightarrow q \equiv \neg(p \wedge \neg q)$$



## Truth Table – And Logic

| p | q | ¬q | p ∧ ¬q | ¬(p ∧ ¬q) | p → q |
|---|---|---|---|---|---|
| T | T | F | F | T | T |
| T | F | T | T | F | F |
| F | T | F | F | T | T |
| F | F | T | F | T | T |



## Analyzing Implication

$$\begin{aligned} p \rightarrow q &\equiv \neg(p \wedge \neg q) \\ &\equiv \neg p \vee q \end{aligned}$$

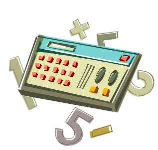


## Logical Operator Precedence

It's just like algebra!

## Logical Operator Precedence

- In *propositional logic*, compound statements can be combined with other statements using logical operators
- So, they can be chained together to form complex logic
- However, just like in algebra, there are possible issues





## Algebra: Order of Operations

- Some mathematical operators have a high "precedence" than others
- They are computed first

```
3 + 6 / 3 * 2
```

## Algebra: Order of Operations

- Knowing the correct order is vital
- For example, what is the result of the expression below?

```
3 + 6 / 3 * 2
```



## Algebra: Order of Operations

- It is 7
- Divide and multiply are equal (and then done left to right), addition is done last

```
3 + 6 / 3 * 2 = 7
```



## Many try, many fail

- Many students, using "PEMDAS" , think multiply is done before divide (M is before D)
- … or they just go left to right with no regard to precedence

```
3 + 6 / 3 * 2 = 4 WRONG! F-!
```



## Standard Precedence Levels

| | |
|---|---|
| 1 | $\neg$ |
| 2 | $\wedge$ |
| 3 | $\vee$ $\oplus$ |
| 4 | $\rightarrow$ |

Highest Level

Lowest Level



## Tautology & Contradiction

When the logic is quite elementary

## Tautology & Contradictions

- Some statements are always true or false regardless of the variables used
- If the statement is always true, it is called *tautology*
- If the statement is always false, it is called a *contradiction*

## Example Tautologies

- The following are examples of tautologies
- The result will always be **true**

```
p ∨ ¬p
p → p
p = p
```



## Example Contradictions

- The following are examples of contradictions
- The result will always be **false**

```
p ∧ ¬p
p ⊕ p
```



## Example

- So, what is the truth table for the example below?
- Let's create a truth table

```
(¬ p ∧ q) ∧ (p ∨ ¬q)
```



## Logic Equivalence

Same meaning, different form

## Logical Equivalence

- *Logical equivalence* is when two different statements are the same
- The truth tables for both statements are identical



## Communitive Law

- Both ∧ and ∨ are communitive
- This means the left-hand and right-hand operands can be switched (symmetric relation)

```
p ∧ q ≡ q ∧ p
p ∨ q ≡ q ∨ p
```

## Idempotent Law

- When a statement is combined with itself, it is equivalent to just the statement (no duplicate)
- This applies to both ∧ and ∨

$$p \land p \equiv p$$
$$p \lor p \equiv p$$



## Involution Law

- One of the most basic equivalences in logic is the *double negation*
- It is fairly obvious, so not more needs to be said

$$\neg \neg p \equiv p$$



## Complement Law

- When a statement is used with its complement (itself negated), it will result in either true or false
- So, it is always as tautology or contradiction

$$p \land \neg p \equiv \text{false}$$
$$p \lor \neg p \equiv \text{true}$$



## Identity Law

- Some operands will have no effect on the truth table of a statement
- In this case, the statement can be simplified

$$p \land \text{true} \equiv p$$
$$p \lor \text{false} \equiv p$$



## Domination Law

- This might look similar to the identity law, but look very careful at which operator is being used
- These will result in either true or false.

$$p \lor \text{true} \equiv \text{true}$$
$$p \land \text{false} \equiv \text{false}$$



## Associative Law

- Some operators in math are *associative*
- For example: (a + b) + c = a + (b + c)
- Same applies to ∧ and ∨

$$p \land (q \land r) \equiv (p \land q) \land r$$

## Distributive Law

- Math has operators that are *distributive*
- For example: a * (b + c) = (a * b) + (a * c)
- Works for both $\land$ and $\lor$

$$p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$$

$$p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$$



## DeMorgan's Law

- *DeMorgan's Law* states important rule for logical equivalency
- These are used to convert And operators to Or and vice-versa



## DeMorgan's Law

- So, it states you can change the operator from $\land$ to $\lor$ or vice-versa
- If you negate both operands

$$\neg (p \land q) \equiv \neg p \lor \neg q$$
$$\neg (p \lor q) \equiv \neg p \land \neg q$$



## Truth Table – Testing Not-Or

| p | q | ¬ p | ¬ q | ¬p ∧ ¬q | ¬ (p ∨ q) |
|---|---|---|---|---|---|
| T | T | F | F | F | F |
| T | F | F | T | F | F |
| F | T | T | F | F | F |
| F | F | T | T | T | T |



## Truth Table – Testing Not-And

| p | q | ¬ p | ¬ q | ¬p ∨ ¬q | ¬ (p ∧ q) |
|---|---|---|---|---|---|
| T | T | F | F | F | F |
| T | F | F | T | T | T |
| F | T | T | F | T | T |
| F | F | T | T | T | T |



## Example

- So, what is the truth table for the example below?
- Let's create a truth table

$$(p \land q) \lor (\neg p \land q)$$

## Boolean Algebra

- However, truth tables become unwieldy as the number of variables increase.
- Logical algebra is another way to evaluate equivalence
- Equivalences can be used to generate one expression from another



## Example Simplification

$(p \land q) \lor (\neg p \land q) =$

$(p \lor \neg p) \land q =$ After using Distributive Law

$\text{true} \land q =$ After using Complement Law

$q$ After using Identify Law



## Arguments

Proving a Point

## Arguments

- A combination of true statements can be used to claim another as true
- An *argument* is a collection of statements (called *premises*), which, when all are true, imply a consequence



## Example Argument

```
raining → wet outside
not wet outside
_____________________
?
```

Our conclusion goes here



## Example Argument 2

```
raining → wet outside
not raining
_____________________
?
```

What conclusion goes here? Does it work?

## Example Argument 3

```
x is duck or x is swan
x isn't a swan
______________________
?
```

Obvious! But why?



## When an Argument is Valid

- When all the premises are true then the consequence must be true
- If all the premises are true, but the conclusion can be false, the argument is disproven



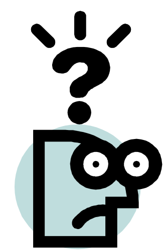
## When an Argument is Valid

- However, if any premise is false, then the argument is not disproven – *it is still valid*
- We can often prove arguments by building truth tables



## Argument Notation

- Arguments can be written out several ways
- The most common approach is to write each premise on a different line
- The consequence is written below the premises separated with horizontal line





## Common Notation

$p \rightarrow q$

$q \rightarrow r$

______

$p \rightarrow r$



## Another Argument Notation

- Arguments can be written on a single line
- Premises are separated with commas
- The consequence follows the symbol $\vdash$ or $\therefore$

"Therefore"

$p \rightarrow q,\ q \rightarrow r \ \vdash \ p \rightarrow r$

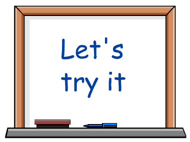
## Let's Try This

$p \to (q \to r)$
$q$

---

$p \to r$

Valid!






## Arguments and Implication

- Arguments are actually implications with each premise $\land$
- So, if you have premises A, premise B, and conclusion C, then it has the following form

$A \land B \to C$



## Valid Arguments

Proving a Point

## Valid Arguments

- *Rules of Inference* are valid arguments that are commonly used in proofs
- Most of these are obvious to you… it is natural logical thought



## Rules of Inference

- *Modus Pones*
  (aka Law of Detachment)
- *Modus Tollens*
- *Disjunctive Syllogism*
- *Hypothetical Syllogism*



## Modus Ponens

$p \to q$
$p$

---

$q$

Rules of Inference

## Modus Ponens Example

If it is a fish, then it lives in water.

It is a fish.

Therefore, it lives in water!



## Modus Tollens

$p \rightarrow q$

$\neg q$

---

$\neg p$

Rules of Inference



## Modus Tollens Example

If it is a fish, then it lives in water.

It doesn't live in water.

Therefore, it is not a fish!



## Disjunctive Syllogism

$p \vee q$

$\neg p$

---

$q$

Rules of Inference



## Disjunctive Syllogism Example

It breathes water or air.

It doesn't breath water.

Therefore, it breathes air.



## Hypothetical Syllogism

$p \rightarrow q$

$q \rightarrow r$

---

$p \rightarrow r$

Rules of Inference

## Hypothetical Syllogism Example

If is a trout, then it is a fish

If it is a fish, then it lives in water.

Therefore, a trout lives in water!



## Let's Apply The Logic

- Let's these rules on the argument below
- We can use either a truth table or logical deduction

```
If I study then I will an A
If I don't watch Netflix then I will study
I didn't get an A
_______________________
I watched Netflix
```



## Simply Logic: letters

- First, let's simply the structure of the argument so we can see the logic
- We will assign each part a letter

```
s = studied
a = got an A
n = watched Netflix
```



## Simply Logic: New Form

1. $s \rightarrow a$
2. $\neg n \rightarrow s$
3. $\neg a$

$\overline{\phantom{xxxxx}}$

$n$



## Modus Tollens – study, no A

1. $s \rightarrow a$
2. $\neg n \rightarrow s$
3. $\neg a$

$\overline{\phantom{xxxxx}}$

$n$

1 and 3:
Modus Tollens



## Modus Tollens – study, no A

1. $\neg s$
2. $\neg n \rightarrow s$
3. $\neg a$

$\overline{\phantom{xxxxx}}$

$n$

1 and 3:
Modus Tollens

"Did not study"

## Modus Tollens – study, Netflix

1. $\neg$ s
2. $\neg$ n $\rightarrow$ s
3. $\neg$ a

---

n

1 and 2:
Modus Tollens



## Modus Tollens – study, Netflix

1. $\neg$ s
2. $\neg$ $\neg$ n
3. $\neg$ a

---

n

1 and 2:
Modus Tollens

**"Did not *not* watch Netflix"**



## Double Negation

1. $\neg$ s
2. $\neg$ $\neg$ n
3. $\neg$ a

---

n

Double negation



## Double Negation

1. $\neg$ s
2. n
3. $\neg$ a

---

n

2:
Double Negation

"Watched Netflix"

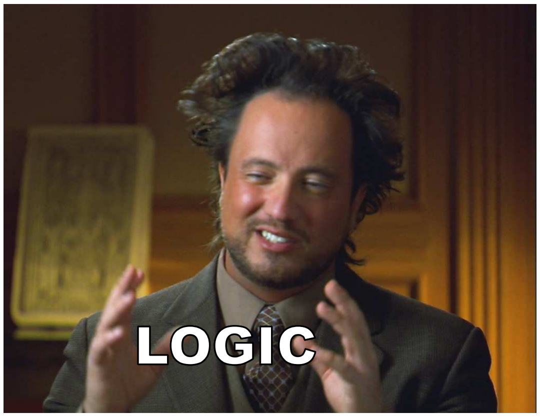


## Logical Fallacies

"This is most illogical" – Mr. Spock

## Logical Fallacies

- There are a number of fallacious arguments that, while they might look logical, are wrong
- The following slides contain some of them
- For fun, apply them to current political discourse or *History Channel 2*




## Fallacy of the Converse

- *Fallacy of the Converse* is based on assumption that if the conclusion is true then the hypothesis is true
- Also called:
  - *affirming the consequent*
  - *converse error*



## Fallacy of the Converse Example

If it is a fish, then it lives in water.

It lives in water.

Therefore, it is a fish!



## Fallacy of the Converse

$p \rightarrow q$
$q$
___
$p$

p can still be false



## Fallacy of the Converse

| p | q | p → q |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |



## Fallacy of the Inverse

- *Fallacy of the Inverse* is based on assumption that if the hypothesis is false, then the conclusion is also false
- Also called:
  - denying the antecedent
  - inverse error

## Fallacy of the Inverse Example

If it is a cat, then it is furry.

It is not a cat.

Therefore, it is not furry!



## Fallacy of the Inverse

$p \rightarrow q$

$\neg p$

---

$\neg q$

q can be either true or false



## Fallacy of the Converse

| p | q | p → q | ¬ p | ¬ q |
|---|---|---|---|---|
| T | T | T | F | F |
| T | F | F | F | T |
| F | T | T | T | F |
| F | F | T | T | T |



## Fallacy of Affirming a Disjunct

- *Fallacy of Affirming a Disjunct* is based on assumption that if there are two attributes and one is true, then the other must be false.
- Other names:
  - *fallacy of the alternative*
  - *false exclusionary disjunct*



## Affirming a Disjunct Example

Suspect is either a politician or a lawyer.

Suspect is a politician.

Therefore, the suspect isn't a lawyer.



## Fallacy of Affirming a Disjunct

$p \vee q$

$p$

---

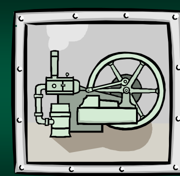
$\neg q$

Just because p is true, doesn't mean q has to be false.

## Fallacy of the Converse

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



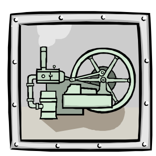


# Boolean Algebra & Proofs

Proofs and Logic are one

## Boolean Algebra

- Remember Boolean algebra laws: Associative, Commutative, etc…
- These can be used to expand an expression… and then simplify it in a different form





## Example

if a or b is true and
a and b is false

then
a = not b



## Example (with a rewrite)

```
We can rewrite it as an argument:

 a ∨ b = true
 a ∧ b = false
 ____________
 a = ¬b
```



## The Strategy

- We *could* use a Truth Table to prove this
- Let's use Boolean Algebra to prove if this is correct

```
a ∨ b = true
a ∧ b = false
____________
a = ¬b
```

## The Approach

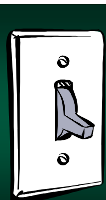
1. Start with a
2. Try to get to ¬ b
3. Use the premises to replace values

```
a ∨ b = true
a ∧ b = false
─────────────
a = ¬b
```



```
a =  a
  =  a  ∧  true               Identity
  =  a  ∧  (b ∨ ¬b)           Complement
  =  a ∧ b    ∨   a ∧ ¬b      Distributive
  =  false    ∨   a ∧ ¬b      Premise
  =  b ∧ ¬b   ∨   a ∧ ¬b      Complement
  = ¬b  ∧  (b ∨ a)            Distributive
  = ¬b  ∧   true              Premise
  = ¬b                        Identity
```



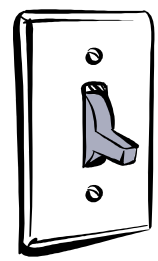


## Duals

Don't "Flip" Out

## Duels

- The dual of a theorem is created by:
  - flip all true and false
  - flip all and all ∨ and ∧
- One interesting thing about Boolean Algebra is that *whenever an argument is true, so is its dual.*





## Example

```
Show p  =  p ∨ p

p  = p
   = p  ∨  false              Identity
   = p  ∨  (p ∧ ¬p)           Complement
   = (p ∨ p) ∧ (p ∨ ¬p)       Distributive
   = (p ∨ p) ∧  true          Complement
   =  p ∨ p                   Identity
```



## Example

```
The dual of this law: p  =  p ∧ p

p  = p
   = p  ∧  true               Identity
   = p  ∧  (p ∨ ¬p)           Complement
   = (p ∧ p) ∨ (p ∧ ¬p)       Distributive
   = (p ∧ p) ∨  false         Complement
   =  p ∧ p                   Identity
```

## Example

```
The dual of this law: p  =  p ∧ p

p  = p
   = p  ∧  true               Identity
   = p  ∧  (p ∨ ¬p)           Complement
   = (p ∧ p) ∨ (p ∧ ¬p)       Distributive
   = (p ∧ p) ∨  false         Complement
   =  p ∧ p                   Identity
```



## Conclusion

- The second proof was simple because all the initial laws were true – as were their duals
- So, any proof of a law *P* using the laws of Boolean algebra can be rewritten using the duals, which would prove the dual of P