Course web: http://www.cs.washington.edu/311

Office hours: 12 office hours each week
Me/James: MW 10:30-11:30/2:30-3:30pm or by appointment

TA Section: Start next week

Call me: Shayan

Don't: Actually call me.

Homework #1: Will be posted today, due next Friday by midnight (Oct 9th)
Gradescope! (stay tuned)

Extra credit: Not required to get a 4.0.
Counts separately.
In total, may raise grade by ~0.1

Don't be shy (raise your hand in the back)!
Do space out your participation.

If you are not CSE yet, please do well!

p	$\neg p$
T	F
F	T

NOT

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

AND

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

OR

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

XOR

$p \to q$

- "If *p*, then *q*" is a **promise**:
  - Whenever *p* is true, then *q* is true
  - Ask "has the promise been broken"

p	q	$p \to q$
F	F	T
F	T	T
T	F	F
T	T	T

If it's raining, then I have my umbrella.

p q

I have my umbrella if it is raining

It is raining only if I have my umbrella

related implications

- Implication: $p \rightarrow q$
- Converse: $q \rightarrow p$
- Contrapositive: $\neg q \rightarrow \neg p$
- Inverse: $\neg p \rightarrow \neg q$

How do these relate to each other?

How to see this?

If raining, I have my umbrella

If I don't have my umbrella then not raining (contrapositive)

$p \leftrightarrow q$

p if q $q \to p$
p only if q $p \to q$
$\to p \Leftrightarrow q$

- *p* iff *q*
- *p* is equivalent to *q*
- *p* implies *q* and *q* implies *p*

p	*q*	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

A fruit is an apple only if it is either red or green and a fruit is not red and green.

p : “Fruit is an apple”
q : “Fruit is red”
r : “Fruit is green”

A fruit is an apple only if it is either red or green and a fruit is not red and green.

↓

(FApple **only if (**FGreen **xor** FRed**)) and** (**not (**FGreen **and** FRed))

↓

(FApple → (FGreen ⊕ FRed)) ∧ (¬ (FGreen ∧ Fred))

↓

p : FApple
q : FGreen
r : FRed

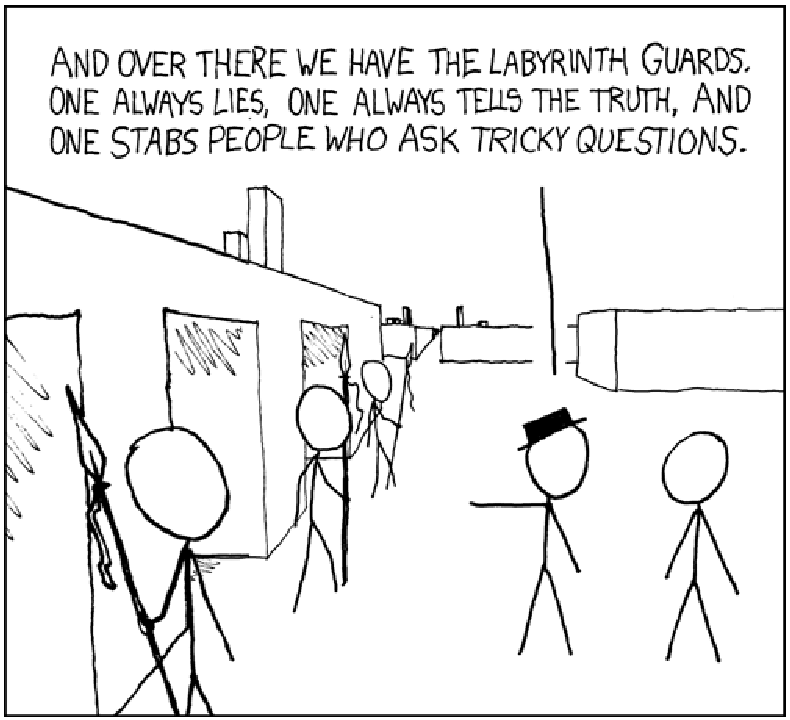
$$(p \rightarrow (q \oplus r)) \wedge (\neg (q \wedge r))$$

Fruit Sentence with a truth table

p	q	r	$q \oplus r$	$p \rightarrow (q \oplus \mathbf{r})$	$q \wedge r$	$\neg(q \wedge r)$	$(p \rightarrow (q \oplus \mathbf{r})) \wedge (\neg(q \wedge r))$
T	T	T					
T	T	F					
T	F	T					
T	F	F					
F	T	T					
F	T	F					
F	F	T					
F	F	F					

Spring 2015

Lecture 2: Digital circuits & more logic



Computing with logic

- **T** corresponds to 1 or “high” voltage
- **F** corresponds to 0 or “low” voltage

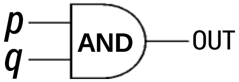
Gates:

- Take inputs and produce outputs (functions)
- Several kinds of gates
- Correspond to propositional connectives

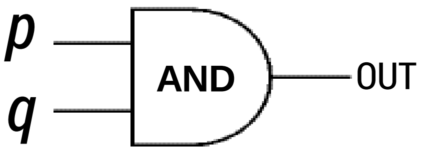
AND Connective vs. AND Gate

$p \wedge q$

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F



p	q	OUT
1	1	1
1	0	0
0	1	0
0	0	0

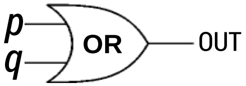


“block looks like D of AND”

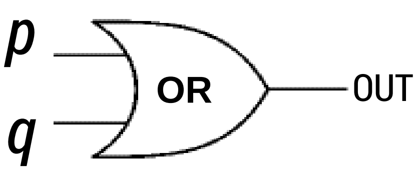
OR Connective vs. OR Gate

$p \vee q$

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F



p	q	OUT
1	1	1
1	0	1
0	1	1
0	0	0

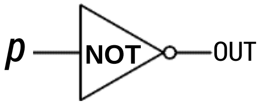


“arrowhead block looks like $\vee$”

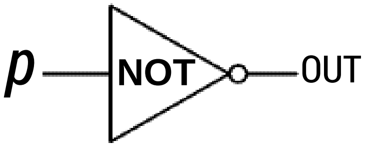
NOT Connective vs. NOT Gate (Also called *inverter*)

$\neg p$

p	$\neg p$
T	F
F	T

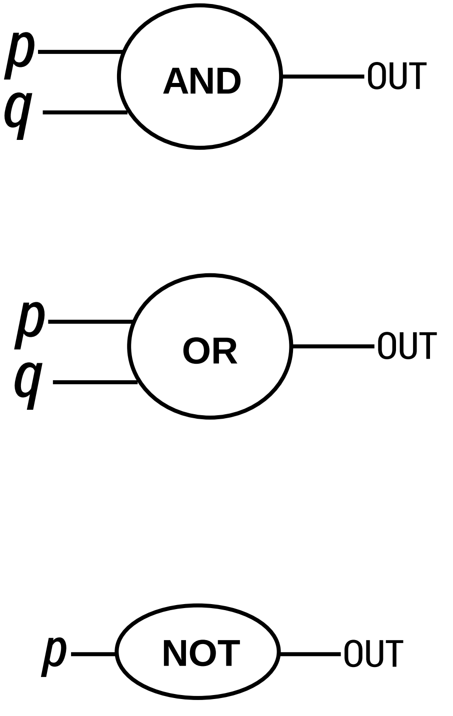


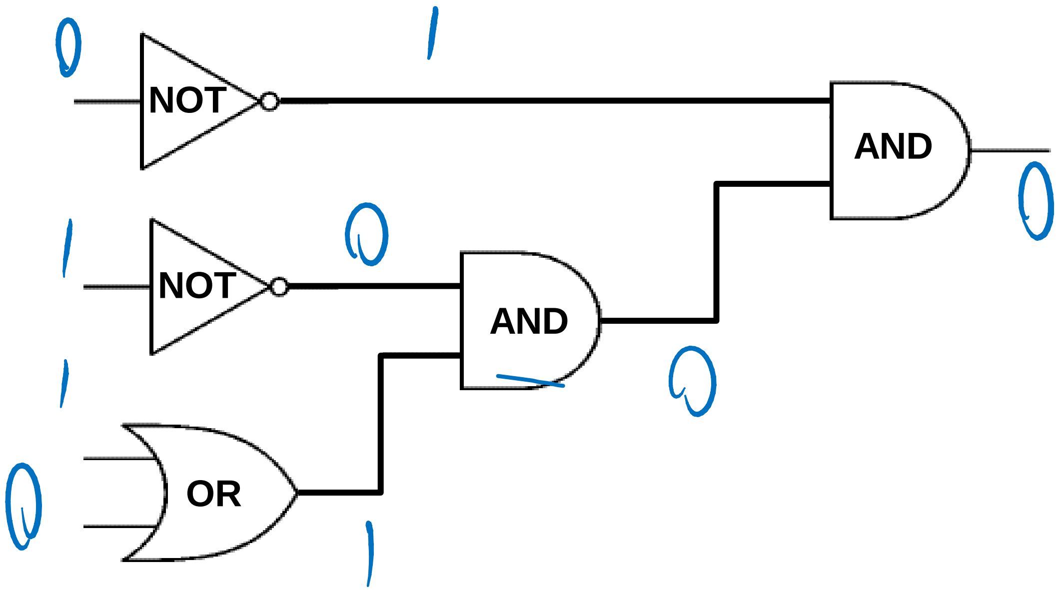
p	OUT
1	0
0	1



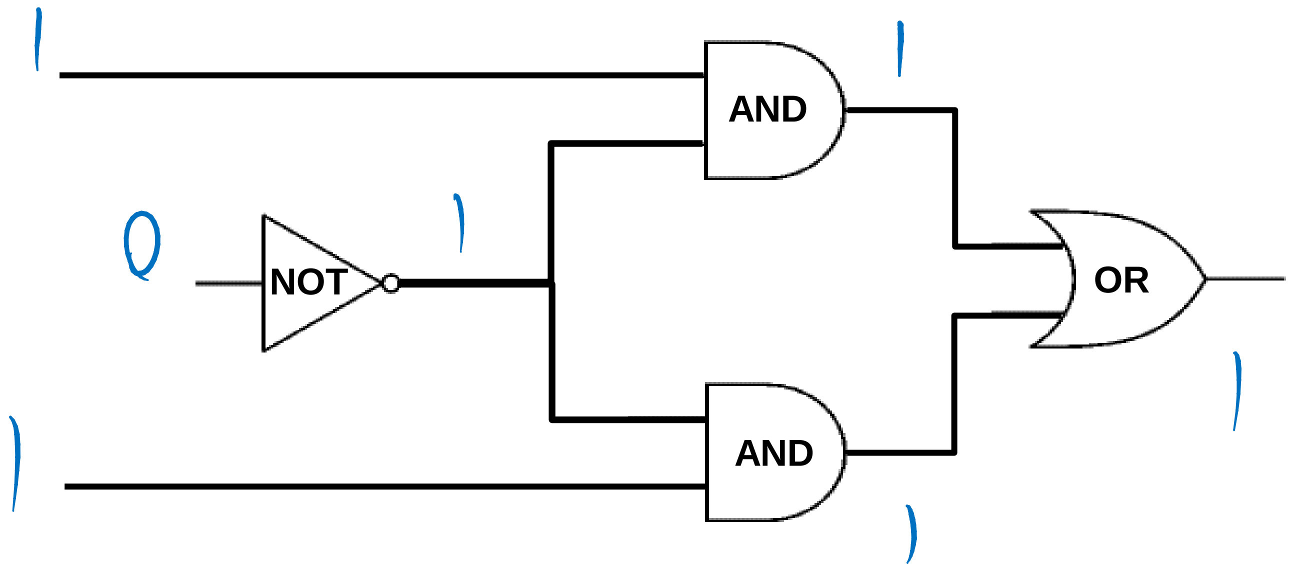
“gee, thanks.”

You can write gates using blobs instead of shapes.





Values get sent along wires connecting gates



Wires can send one value to multiple gates!

Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

Classify!

$p \vee \neg p$ Tautology

$p \oplus p$ Contradiction

$(p \rightarrow q) \wedge p$ Contingency

$(p \wedge q) \vee (p \wedge \neg q) \vee (\neg p \wedge q) \vee (\neg p \wedge \neg q)$ Tautology

T T T F F T F F

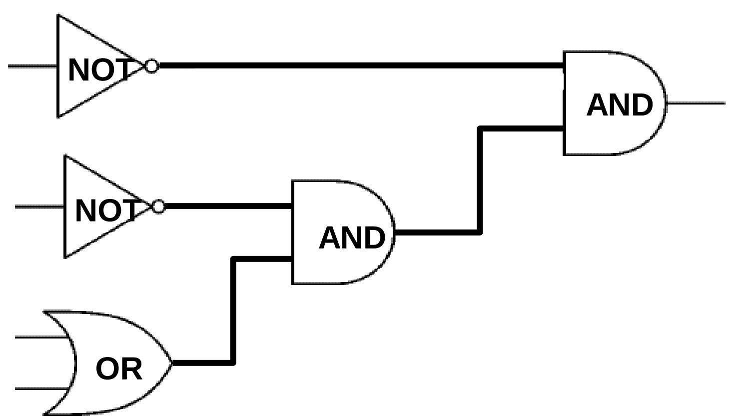
Terminology: A compound proposition is a…

- *Tautology* if it is always true
- *Contradiction* if it is always false
- *Contingency* if it can be either true or false

Classify!

Sudoku!

$$((p \wedge q \wedge r) \vee (\neg p \wedge q \wedge \neg r)) \wedge ((p \vee q \vee \neg s) \vee (p \wedge q \wedge s))$$



P != NP

A and B are *logically equivalent* if and only if

$A \leftrightarrow B$ is a tautology

i.e. A and B have the same truth table

The notation $A \equiv B$ denotes A and B are logically equivalent.

Example: $p \equiv \neg\neg p$

p	$\neg p$	$\neg\neg p$	$p \leftrightarrow \neg\neg p$
T	F	T	T
F	T	F	T

$A \leftrightarrow B$ vs. $A \equiv B$

$A \equiv B$ says that ***two*** propositions A and B *always* ***mean*** *the same thing.*

$A \leftrightarrow B$ is a ***single*** proposition that may be true or false depending on the truth values of the variables in A and B.

but $A \equiv B$ and $(A \leftrightarrow B) \equiv \mathbf{T}$ have the same meaning.

Note: Why write $A \equiv B$ and not $A=B$?

[We use $A=B$ to say that A and B are precisely the same proposition (same sequence of symbols)]

$p = p$

$p \neq \neg\neg p$

$p \equiv \neg\neg p$

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

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

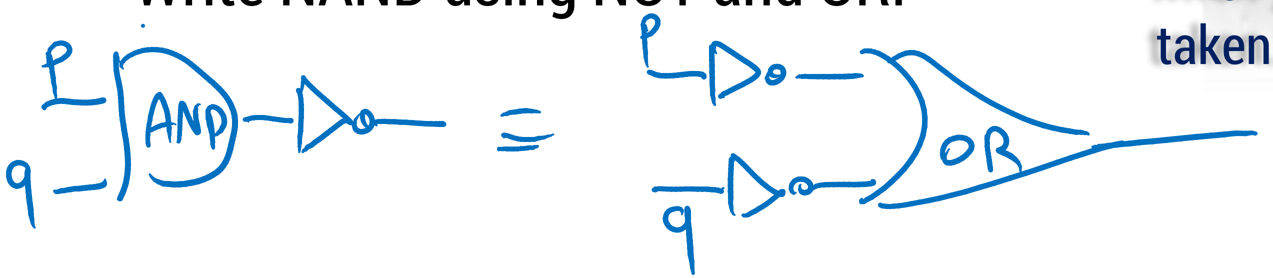
"Always wear breathable fabrics when you get your picture taken."

My code compiles or there is a bug.

[let's negate it]

My code does not compile and there is no bug

Write NAND using NOT and OR:



Verify: $\neg(p \wedge q) \equiv (\neg p \vee \neg q)$

p	q	$\neg p$	$\neg q$	$\neg p \vee \neg q$	$p \wedge q$	$\neg(p \wedge q)$	$\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$
T	T	F	F	F	T	F	T
T	F	F	T	T	F	T	T
F	T	T	F	T	F	T	T
F	F	T	T	T	F	T	T

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

```
if !(front != null && value > front.data)
    front = new ListNode(value, front);
else {
    ListNode current = front;
    while !(current.next == null || current.next.data >= value)
        current = current.next;
    current.next = new ListNode(value, current.next);
}
```

If Case: front == null or value ≤ front.data

While stops: current.next == null or current.next.data ≥ value

Repeated calls give a sorted linked list.

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

p	q	$p \rightarrow q$	$\neg p$	$\neg p \vee q$	$(p \rightarrow q) \leftrightarrow (\neg p \vee q)$
T	T	T	F	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

If it is raining then I have my umbrella

It is not raining or I have my umbrella.

Describe an algorithm for computing if two logical expressions/circuits are equivalent.

What is the run time of the algorithm?

- $x + y = y + x$ (commutativity)
- $x \cdot (y + z) = x \cdot y + x \cdot z$ (distributivity)
- $(x + y) + z = x + (y + z)$ (associativity)

Logic has similar algebraic properties

- $x + y = y + x$ (commutativity)
  - $p \lor q \equiv q \lor p$
  - $p \land q \equiv q \land p$

- $x \cdot (y + z) = x \cdot y + x \cdot z$ (distributivity)
  - $p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$
  - $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$

- $(x + y) + z = x + (y + z)$ (associativity)
  - $(p \lor q) \lor r \equiv p \lor (q \lor r)$
  - $(p \land q) \land r \equiv p \land (q \land r)$

You will always get this list.

- Identity
  - $p \wedge \mathrm{T} \equiv p$
  - $p \vee \mathrm{F} \equiv p$
- Domination
  - $p \vee \mathrm{T} \equiv \mathrm{T}$
  - $p \wedge \mathrm{F} \equiv \mathrm{F}$
- Idempotent
  - $p \vee p \equiv p$
  - $p \wedge p \equiv p$
- Commutative
  - $p \vee q \equiv q \vee p$
  - $p \wedge q \equiv q \wedge p$

- Associative

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

  $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
- Distributive

  $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

  $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$
- Absorption

  $p \vee (p \wedge q) \equiv p$

  $p \wedge (p \vee q) \equiv p$
- Negation

  $p \vee \neg p \equiv \mathrm{T}$

  $p \wedge \neg p \equiv \mathrm{F}$