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!
### Logical Connectives

#### NOT

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\neg p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

#### AND

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \land q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

#### OR

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \lor q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

#### XOR

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \oplus q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
• “If \( p \), then \( q \)” is a **promise**:
  • Whenever \( p \) is true, then \( q \) is true
  • Ask “has the promise been broken”

\[
\begin{array}{|c|c|c|}
\hline
p & q & p \rightarrow q \\
\hline
F & F & T \\
F & T & T \\
T & F & F \\
T & T & T \\
\hline
\end{array}
\]

\[
\begin{array}{l}
\text{If it’s raining, then I have my umbrella.}
\end{array}
\]

\[
\begin{array}{l}
\text{(I have my umbrella if it is raining)}
\end{array}
\]

\[
\begin{array}{l}
\text{(It is raining only if I have my umbrella)}
\end{array}
\]
• 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 rainy, I have my umbrella
If I don't have my umbrella then not raining (contrapositive)
• $p$ iff $q$
• $p$ is equivalent to $q$
• $p$ implies $q$ and $q$ implies $p$

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \iff q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>
A fruit is an apple only if it is either red or green and a fruit is not red and green.

\[ p : \text{“Fruit is an apple”} \]
\[ q : \text{“Fruit is red”} \]
\[ r : \text{“Fruit is green”} \]
Let's think about fruits

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

\[(\text{FApple only if (FGreen xor FRed)) and (not (FGreen and FRed))}\]

\[(\text{FApple } \rightarrow (\text{FGreen } \oplus \text{FRed })) \land ( \neg (\text{FGreen } \land \text{FRed}))\]

\[p : \text{FApple}\]
\[q : \text{FGreen}\]
\[r : \text{FRed}\]

\[((p \rightarrow (q \oplus r)) \land (\neg (q \land r)))\]
Fruit Sentence with a truth table

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$r$</th>
<th>$q \oplus r$</th>
<th>$p \rightarrow (q \oplus r)$</th>
<th>$q \land r$</th>
<th>$\neg (q \land r)$</th>
<th>$(p \rightarrow (q \oplus r)) \land (\neg (q \land r))$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
AND OVER THERE WE HAVE THE LABYRINTH GUARDS. ONE ALWAYS LIES, ONE ALWAYS TELLS THE TRUTH, AND ONE STABS PEOPLE WHO ASK TRICKY QUESTIONS.
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

#### AND Gate

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### AND Connective

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$p \wedge q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

“block looks like D of AND”
OR gate

OR Connective vs. OR Gate

\[ p \lor q \]

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>( p \lor q )</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( p )</th>
<th>( q )</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

“arrowhead block looks like \( \lor \)”
NOT Connective

$$\neg p$$

<table>
<thead>
<tr>
<th>$p$</th>
<th>$\neg p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

vs.

NOT Gate (Also called inverter)

<table>
<thead>
<tr>
<th>$p$</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
You can write gates using blobs instead of shapes.

```
p  AND  OUT
q

p  OR  OUT
q

p  NOT  OUT
```
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!**

\[
p \lor \neg p
\]

\[
p \oplus p
\]

\[
(p \rightarrow q) \land p
\]

\[
(p \land q) \lor (p \land \neg q) \lor (\neg p \land q) \lor (\neg p \land \neg q)
\]
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!

\[
((p \land q \land r) \lor (\neg p \land q \land \neg r)) \land ((p \lor q \lor \neg s) \lor (p \land q \land s))
\]

\[P \neq NP\]
A and B are *logically equivalent* if and only if

\[ A \iff B \] 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:** \( p \equiv \neg \neg p \)

<table>
<thead>
<tr>
<th>p</th>
<th>\neg p</th>
<th>\neg \neg p</th>
<th>p \iff \neg \neg p</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
A ⇔ B vs. A ≡ B

A ≡ B says that two propositions A and B always mean the same thing.

A ⇔ 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 ≡ B and (A ⇔ B) ≡ T have the same meaning.

Note: Why write A ≡ B and not A = B?
[We use A = B to say that A and B are precisely the same proposition (same sequence of symbols)]
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:

\[
\neg (p \land q) \equiv \neg p \lor \neg q
\]

\[
\neg (p \lor q) \equiv \neg p \land \neg q
\]

“Always wear breathable fabrics when you get your picture taken.”
Verify: \( \neg(p \land q) \equiv (\neg p \lor \neg q) \)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>(\neg p)</th>
<th>(\neg q)</th>
<th>(\neg p \lor \neg q)</th>
<th>(p \land q)</th>
<th>(\neg (p \land q))</th>
<th>(\neg (p \land q) \leftrightarrow (\neg p \lor \neg q))</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
\[ \neg (p \land q) \equiv \neg p \lor \neg q \]
\[ \neg (p \lor q) \equiv \neg p \land \neg q \]

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
Repeat until calls give a sorted linked list.
law of implication

\[(p \rightarrow q) \equiv (\neg p \lor q)\]

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>p → q</th>
<th>¬p</th>
<th>¬p ∨ q</th>
<th>(p → q) ↔ (¬p ∨ q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

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?
some familiar properties of arithmetic

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

Logic has similar algebraic properties
some familiar properties of arithmetic

• $x + y = y + x$  
  \[- p \lor q \equiv q \lor p \]
  \[- p \land q \equiv q \land p \]

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

• $(x + y) + z = x + (y + z)$  
  \[- (p \lor q) \lor r \equiv p \lor (q \lor r) \]
  \[- (p \land q) \land r \equiv p \land (q \land r) \]
properties of logical connectives

- **Identity**
  - $p \land T \equiv p$
  - $p \lor F \equiv p$

- **Domination**
  - $p \lor T \equiv T$
  - $p \land F \equiv F$

- **Idempotent**
  - $p \lor p \equiv p$
  - $p \land p \equiv p$

- **Commutative**
  - $p \lor q \equiv q \lor p$
  - $p \land q \equiv q \land p$

- **Associative**
  - $(p \lor q) \lor r \equiv p \lor (q \lor r)$
  - $(p \land q) \land r \equiv p \land (q \land r)$

- **Distributive**
  - $p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$
  - $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$

- **Absorption**
  - $p \lor (p \land q) \equiv p$
  - $p \land (p \lor q) \equiv p$

- **Negation**
  - $p \lor \neg p \equiv T$
  - $p \land \neg p \equiv F$

You will always get this list.