Let’s use OOP to simulate something.
A person has a name.
A person can be a safety officer.
A person can have at most one boss.

We can access a person's name.
We can make someone be a safety officer.
We can determine whether a person is a safety officer.
We can access a person's boss.

Maria is Lisa’s boss.
Lisa is Melissa's boss.
Lisa is Theresa’s boss.
Melissa is Ben’s boss.
Melissa is Colleen’s boss.
Theresa is Brian’s boss.
Theresa is John’s boss.
Brian is a safety officer.
A person has a name.

```java
public class Person {
    private String name;

    public Person(String name) {
        this.name = name;
    }

    // getters...
    // auto-generated hashCode and equals...
}

Person ben = new Person("Ben");
System.out.println(ben.getName());
```
A person can be a safety officer.

```java
Person brian = new Person("Brian");
brian.makeSafetyOfficer();

public class Person {
    ...
    private boolean isSafetyOfficer;
    ...
    public Person(String name) {
        this.name = name;
        this.isSafetyOfficer = false;
    }
    ...
    public void makeSafetyOfficer() {
        this.isSafetyOfficer = true;
    }
    ...
}
```
A person can have at most one boss.

Person lisa = new Person("Lisa", maria);
System.out.println(lisa.getBoss().getName());
System.out.println(maria.getBoss());

```java
public class Person {
    ...
    private Person boss;
    ...
    public Person(String name) {
        this.name = name;
        this.isSafetyOfficer = false;
        this.boss = null;
    }

    public Person(String name, Person boss) {
        this(name);
        this.boss = boss;
    }
    ...
    public Person getBoss() {
        return this.boss;
    }
    ...
```
A person has a name.
A person can be a safety officer.
A person can have at most one boss.

We can access a person's name.
We can make someone be a safety officer.
We can determine whether a person is a safety officer.
We can access a person's boss.

Does A work with B? (i.e., are A and B coworkers?)
A and B are coworkers if and only if they have the same boss.

Is B the employee of A?
B is the employee of A if and only if A is B’s boss.

Who are A’s employees?
A’s employees are all the people whose boss is A.

Who are A's safety officers?
A’s safety officers are any coworkers of A who are safety officers.
Does A work with B?

A works with B if and only if they have the same boss.

```
ben.worksWith(colleen)
ben.worksWith(brian)
```

```java
public boolean worksWith(Person person) {
    Person myBoss = this.getBoss();
    Person theirBoss = person.getBoss();
    return myBoss != null &&
           myBoss.equals(theirBoss);
}
```
Why can’t we just say this?

Maria is Lisa’s boss.
Lisa is Melissa's boss.
Lisa is Theresa’s boss.
Melissa is Ben’s boss.
Melissa is Colleen’s boss.
Theresa is Brian’s boss.
Theresa is John’s boss.
Brian is a safety officer.

Person A is an employee of person B if B is A's boss.

Person A works with person B if they have the same boss.

Person A’s safety officer is anyone A works with who is a safety officer.
Behold: Prolog!

boss(maria, lisa).
boss(lisa, melissa).
boss(lisa, theresa).
boss(melissa, ben).
boss(melissa, colleen).
boss(theresa, brian).
boss(theresa, john).
safetyOfficer(brian).

employee(PersonA, PersonB) :- boss(PersonB, PersonA).

worksWith(PersonA, PersonB) :-
    boss(Boss, PersonA), boss(Boss, PersonB).

safetyOfficer(PersonA, PersonB) :-
    worksWith(PersonA, PersonB), safetyOfficer(PersonA).
1.1 Syntax of the Predicate Calculus

4.1 Definition. The syntax of the predicate calculus ($\mathcal{P}C$) consists of symbols and formulas as follows:

Symbols
- parentheses: (, )
- sentential connectives: $\neg$, $\lor$, $\land$, $\rightarrow$, $\leftrightarrow$
- quantifiers: $\forall$, $\exists$

Sentential letters (sentential letters): $A, B, \cdots Z$, and any of these letters with a positive Arabic numeral subscript.

Predicate symbols: An $n$-ary predicate is an uppercase letter, $A, \cdots Z$, with the numeral $n$ as a superscript, where $n$ denotes the arity of the predicate and $0 < n$. These uppercase letters may also have numerical subscripts. Note: We will usually omit the superscript when we know the arity of a predicate.

Individual constants: lowercase letters $a, \cdots r$, with or without numerical subscripts.

Individual variables: lowercase letters $s, \cdots z$, with or without numerical subscripts.

Formulas
The set of all predicate calculus ($\mathcal{P}C$) formulas is defined recursively, beginning with the atomic formulas.

Atomic Formula:
Any single $\mathcal{S}C$ letter, or an $n$-ary predicate followed by exactly $n$ symbols, each of which is either an individual constant or a variable.

Formula:
Any atomic formula, or any expression (finitely long string of symbols) that is obtainable by use of the following predicate calculus construction rules (PCCR):
Prolog is syntactic sugar for the predicate calculus.*
Hmmm  ≡  Racket  ≡  Prolog

\[ 0 \text{ jumpn } 0 \]

\[ (\lambda (x) (x \ x)) (\lambda (x) (x \ x)) \]

\[ p :- p \]
1. Language influences thought.

2. Programs $\equiv$ Data
   and self-reference is powerful

3. Computers are powerful.
   they change our lives for good and bad
Commentary: Facebook’s Algorithm vs. Democracy

By Cathy O'Neil on Wed, 07 Dec 2016

Over the last several years, Facebook has been participating—unintentionally—in the erosion of democracy.

The social network may feel like a modern town square, but thanks to its tangle of algorithms, it's nothing like the public forums of the past. The company determines, according to its interests and those of its shareholders, what we see and learn on its social network. The result has been a loss of focus on critical national issues, an erosion of civil disagreement, and a threat to democracy itself.

Facebook is just one part—though a large part—of the Big Data economy, one...
Agile 3D Sketching with Air Scaffolding

Tongkwan Kim, Sang-Gyu Ahn, Joon Hyub Lee, Seok-Hyung Bae
Machinations Calcite

Clark Coolidge

acetone-imprinted

acetone swatch on the car barn oil wall
ocarina & mumps
function wet green
I'd leave sole key to this game to my friend, shoe water car

octopus impressed
weaving candle turn on computer cigarette paper wall tar heels & balance
a lot of yellow hitch cock

He'll have to hurry & carry away, to my blue friend hustling bringing his moon & car

gate inked
merry melodies drool on canvas of wet lead star tool crayon & sands
length of granite acc. - drill
It's sucking up the strand, his crystal flag, & the eels tube for their parade swizzle fun

action tex
spiller dry - ice spazzles ing ace supper at church
hard pinks & sponge breath
many forarms drift

Roller window going up on I repeat my offer food list in iron flakes