What can be programmed?

- **What class of problems can be solved?**
  - G5, 1000Mhz Pentium III, Cray, pencil?
  - Alan Turing proved some things, hypothesized others
    - Halting problem, Church-Turing thesis

- **What class of problems can be solved efficiently?**
  - Problems with no practical solution
    - What does practical mean?
  - Problems for which we can’t find a practical solution
    - Solving one solves them all
    - Would you rather be rich or famous?
Schedule students, minimize conflicts

- Given student requests, available teachers
  - write a program that schedules classes
  - Minimize conflicts

- Add a GUI too
  - Web interface
  - ...
  - ...

I can’t write this program because I’m too dumb
One better scenario

I can’t write this program because it’s provably impossible
Another possible scenario

I can’t write this program but neither can all these famous people
The halting problem: writing \texttt{doesHalt}

```java
public class ProgramUtils {
    /**
     * Returns true if programe halts on input,
     * otherwise returns false (programe loops)
     */
    public static boolean doesHalt(String programe, String input) {
    }
}
```

- A compiler is a program that reads other programs as input
  - Can a word counting program count its own words?
- The \texttt{doesHalt} method might simulate, analyze, ...
  - One program/function that works for \textit{any} program/input
public static void main(String[] args) {
    String prog = "Foo.java";
    String input = "123 456"
    if (ProgramUtils.doesHalt(prog,input)) {
        System.out.println(prog+" stops");
    }
    else {
        System.out.println(prog+" 4ever");
    }
}

● Can user enter name of program? Input?
  ➢ What's the problem with this program?
Consider the class *Confuse.java*

```java
public static void main(String[] args) {
    String prog = "Foo.java";
    if (ProgramUtils.doesHalt(prog, prog)) {
        while (true) {
            // do nothing forever
        }
    }
}
```

- **We want to show writing `doesHalt` is impossible**
  - Proof by contradiction:
  - Assume possible, show impossible situation results

- **Can a program read a program? Itself?**
What's a meta catalog? Top 10 sites?

- Consider a website of interesting sites
  - Does the website list itself? Is this a problem?

- Consider a website that lists every useless website
  - Would this be a useful resource?
  - Does the website list itself?

- What about a site of all the sites that list themselves?
  - What about sites that don't list themselves? nolist.com
Not impossible, but impractical

- **Towers of Hanoi**
  - How long to move n disks?

- **What combination of switches turns the light on?**
  - Try all combinations, how many are there?
  - Is there a better way?
**Travelling Salesperson**

- Visit every city exactly once
- Minimize cost of travel or distance
- Is there a tour for under $2,000? less than 6,000 miles?
- Is close good enough?
  - Within 10% of optimal
  - Within 50% of optimal
  - …

Try all paths, from every starting point -- how long does this take?

a, b, c, d, e, f, g
b, a, c, d, e, f, g …
Are hard problems easy?

- **P = easy problems, NP = “hard” problems**
  - P means solvable in polynomial time
    - Difference between $N$, $N^2$, $N^{10}$?
  - NP means non-deterministic, polynomial time
    - guess a solution and verify it efficiently

- **Question: P = NP?**
  - if yes, a whole class of difficult problems, the NP-complete problems, can be solved efficiently
  - if no, none of the hard problems can be solved efficiently
  - showing the first problem was NP complete was an exercise in intellectual bootstrapping, satisfiability/Cook/(1971)
Theory and Practice

- **Number theory: pure mathematics**
  - How many prime numbers are there?
  - How do we factor?
  - How do we determine primeness?

- **Computer Science**
  - Primality is “easy”
  - Factoring is “hard”
  - Encryption is possible

- Top secret
- public-key cryptography
- randomized primality testing