LWoC

- Review Recommender, dictionaries, files
  - How to create recommendations in order? food.txt
  - Toward a Duke eatery-recommender system, tools?

- Test2, reworked, grades, more
  - The good news, the regular news, the fiscal cliff, ...

- Limits of Computation
  - Can you do anything with enough knowledge and time?
  - Why?

What is Computing? Informatics?

- What is computer science, what is its potential?
  - What can we do with computers in our lives?
  - What can we do with computing for society?
  - Will networks transform thinking/knowing/doing?
  - Society affecting and affected by computing?
  - Changes in science: biology, physics, chemistry, ...
  - Changes in humanity: access, revolution (?), ...

- Privileges and opportunities available if you know code
  - Writing and reading code, understanding algorithms
  - Majestic, magical, mathematical, mysterious, ...

What can be programmed?

- What class of problems can be solved?
  - Hadoop, Cloud, Mac, Windows8, Android, ...
  - Alan Turing contributions
    - Halting problem, Church-Turing thesis

- What class of problems can be solved efficiently?
  - Problems with no practical solution
    - What does practical mean?
  - 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
I can’t write this program but neither can all these famous people.

One better scenario?

Still another scenario, is this better?

Summary of Problem Categories

- Some problems can be solved 'efficiently'
  - Run large versions fast on modern computers
  - What is 'efficient'? It depends

- Some problems cannot be solved by computer.
  - Provable! We can’t wait for smarter algorithms

- Some problems have no efficient solution
  - Provably exponential $2^n$ so for "small" $n$ ...

- Some have no known efficient solution, but ...
  - If one does they all do!

Entscheidungsproblem

- What can we program?
  - What kind of computer?

- What can’t we program?
  - Can't we try harder?

- Can we write a program that will determine if any program $P$ will halt when run on input $S$?
  - Input to halt: $P$ and $S$
  - Output: yes/no halts

Good sites: http://del.icio.us/

- What is social bookmarking?
  - Why is del.icio.us interesting?
  - Who posts, who visits?

- What about a website of interesting websites?
  - What would you expect to find there?
  - Would the site list itself?

- What about sites that list/link to themselves?
  - What about a site with all sites that list themselves?
Bad sites: http://haz.ardo.us

- Sites listing bad sites (don’t visit them?)
  - Websites of all the sites that don’t list themselves?
    - Is notlisted.com listed on notlisted.com?
  - Websites of all the sites that don’t list themselves?

halting module/problem: writing doesHalt

```python
# function doesHalt returns True if programe halts when run on input, and False if program doesn't halt (infinite loop)

# def doesHalt(programe,input):
#     # code here

name = "SpreadingNews.py"
data = "input.txt"
if doesHalt(name,data):
    print "program ended!"
else:
    print "loops 4ever"
```

We want to show writing doesHalt is impossible

Proof by contradiction:

Assume possible, show impossible situation results

Can a program read a program? Itself?

How to tell if X stops/halts on Y

```
import halting
def runHalt():
    prog = "SpreadingNews.py"
    input = ["abc", "def", "hij"]
    if halting.doesHalt(prog,input):
        print prog,"stops"
    else:
        print prog,"loops 4ever"
```

Consider this module Confuse.py

```
import halting
print "enter name of program",
prog = raw_input()
if halting.doesHalt(prog,prog):
    while True:
        pass
    print "finished"
```

We want to show writing doesHalt is impossible

Proof by contradiction:

Assume possible, show impossible situation results

Can a program read a program? Itself?
Some problems take forever, but …

- Can we visit all cities, no repeats, using Southwest, for less than $123,329.50
  - RDU->MCO->...->...->...->DEN
  - RDU->DEN->...->...->...->MCO
  - repeat and test, what's the issue here?
  - Can we find shortest path for packets on Internet? Yes!
  - Can we find longest path for silent meditation? No!
  - We don't know how, but if we did!!!

- Contrast towers of Hanoi, $2^n$ moves always!

Are hard problems easy? Clay Prize

- $P = easy$ problems, $NP = hard$ problems
- $P$ means solvable in polynomial time
- $NP$ 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, no hard problems can be solved efficiently
  - showing the first problem was $NP$-complete was an exercise in intellectual bootstrapping, satisfiability/Cook/(1971)

How is Python like all other programming languages, how is it different?

- Why do we use [Python | Java] in courses?
  - [is | is not] Object oriented
  - Large collection of libraries
  - Safe for advanced programming and beginners
  - Harder to shoot ourselves in the foot

- Why don't we use C++ (or C)?
  - Standard libraries weak or non-existant (comparatively)
  - Easy to make mistakes when beginning
  - No GUIs, complicated compilation model
  - What about other languages?