Level 3 CS: what I think worked (I hope)

Dr N Leslie
Onslow College

Hakihea December 2015

Onslow College

Topics

- formal languages
- ~~network communication protocols~~
- complexity and tractability
- intelligent systems
- software engineering
- ~~graphics and visual computing~~

Key points for students

- Why is this a computer science problem?
- Show me some examples – why do we care?
- Preferably, show me some code, but at least show something working

Key points for me

- Can I make it accessible?
- Can I make it relevant?
- Can I promote student voice?

Sources

- Student experiences
- CS Field guide
- Code from e.g. Google
- Student investigations
- . . .

Software engineering

What are the issues?

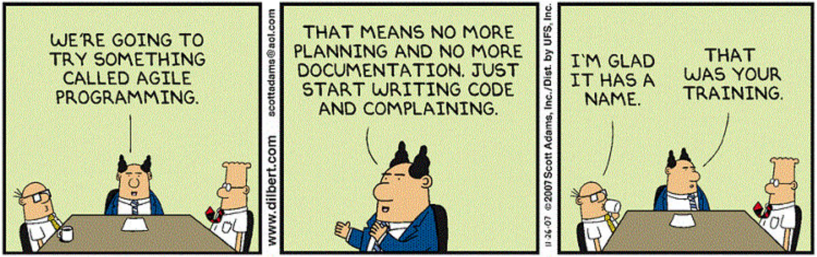
- "Software crisis"
- Ariane
- Novopay
- Therac25
- and many, many others

Not all bad – good Software Engineers get very well paid.

Student experiences

- Students have written programs. . .
- . . . but only very small ones
- Reflect on L1/L2 vs L3 programming standards (Waterfall vs Agile?)
- Model Agile Methods in L3 programming

Introducing Agile


WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
www.dilbert.com scottadams@aol.com
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
11-26-07 ©2007 Scott Adams, Inc./Dist. by UFS, Inc.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.

Use Post-it Notes and videos

Intelligent Systems

"Why is this a computer science problem?"

- Introduce Imitation Game/Turing Test – avoids philosophical (read "pointless") debates.
- Let students find own examples/point out 'obvious' things:
  - ELIZA & chatbots
  - Minimax in games
  - Google translate – how does it work, where does it fail?
  - Tiny Comp. Ling. example from L2 – compute plurals

Lots of AI problems are easy for people, but hard for machines.

Game playing – Minimax

- Choose the worst best choice for your opponent
- Use noughts and crosses as an example
- Even this is (probably) to hard for students to code, but students can explain
- May be relateable to economics/other subjects

Onslow College

Natural language – Google translate

Google translate: how does it work? Why does it fail?

- Example of translation of e.g. Le Monde
- However:
  - *She is very beautiful* becomes *Elle est très belle*
  - *She is very, very beautiful* becomes **Elle est très, très beau*

Something simpler – compute plurals

Actually from an example in a post-grad text in computational linguistics!

- potato → potatoes
- photo → photos
- portico → porticos (although my spelling checker disagrees!)

Students can explain the code and the need for background knowledge/intelligence.

Natural Language – ELIZA

- The mother of all chatbots
- Students can use chatbots – not always a good idea to watch interactions
- How does it work?

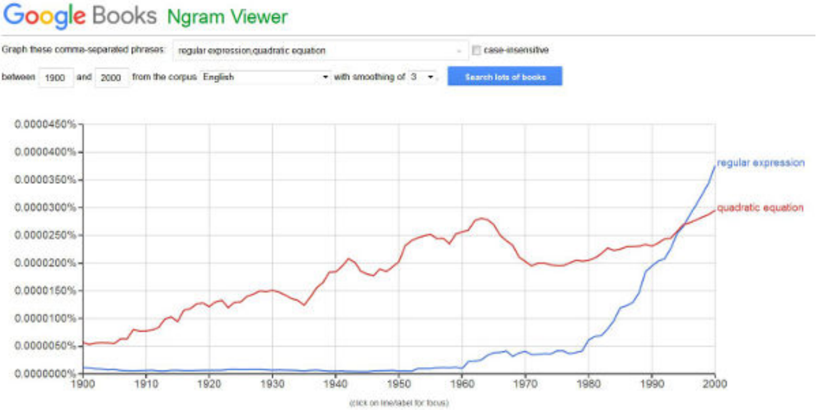
Chatbot code adapted from Google

```
def chat():
    question = input().lower()
    while not("bye!" in  question):
        if 'who' in question:
            print('My name is StupidBot.')
        elif 'you are' in question:
            print('Why do you say I am' + (
               question.split('you are'))[-1]
               +"?")
        else:
            print("I don't understand. Ask me
               another question.")
        question = input().lower()
```

Formal Languages

- Students do not have the language to talk of symbols, strings, words, alphabets, languages (What do they get taught in maths nowadays?)
- Have no idea what a finite automaton is, nor what a grammar is. . .
- May have seen regular expression searches, e.g. Notepad++, grep (ha!)
- Factoid: The term "Regular expression" was coined in the 1950s.

Should you care?



Google Books Ngram Viewer

Graph these comma-separated phrases: regular expression,quadratic equation □ case-insensitive

between 1900 and 2000 from the corpus English with smoothing of 3

Where do we find formal languages?

- C, Java, Python, etc.
- Key presses to use an ATM
- Key presses to use a microwave
- Mouse-clicks & key presses to use software
- Many, many problems in computer science can be expressed as problems in formal language theory (often naturally)

What seemed to work for my students

- Need some background/vocabulary.
- Grammar as generator/automaton (or program) as acceptor.
- Chomsky hierarchy
- Regular languages very easy to describe/recognise

Uses of regular expressions

- Regular expressions in text search
- Describe valid variable names in your favourite programming language, or Python
- Describe UI in terms of key-presses

Summary

- I tried to link Software Engineering to student experience
- I tried to cut topics down to size
- I encourage students to use CS Field Guide (rather than Wikipedia)
- I only allowed students to do certain topics that I was happy to supervise

Onslow College