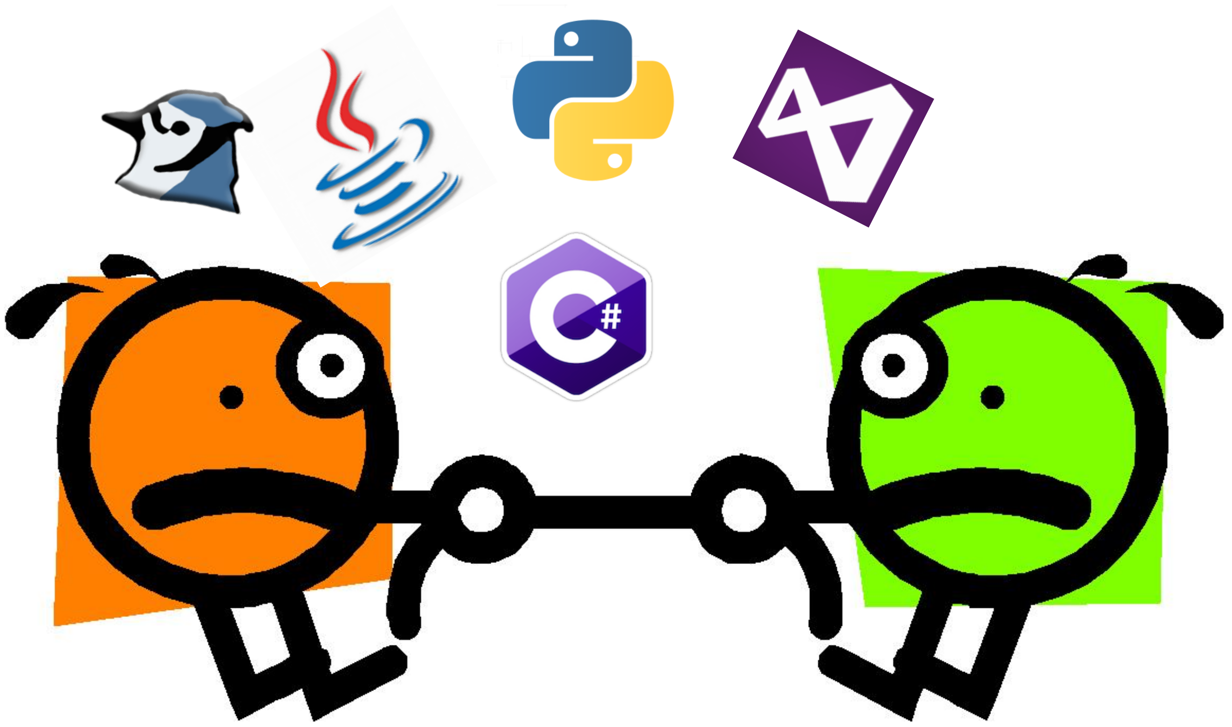
A Longitudinal Study of Introductory Programming Courses in Australasia

Raina Mason, Southern Cross University

Simon, University of Newcastle

A longitudinal study

- What programming languages are used in introductory programming courses in Australia & New Zealand?
- What IDEs/environments are used with them, and why?
- Debate ….



A longitudinal study

- Aim ***assumed to be to*** maximise success and motivation and prepare students for careers.
- Range of reasons – do these change over time?
- Several other questions
  - de Raadt, Watson, Toleman: 2001
  - de Raadt, Watson, Toleman: 2003
  - Mason, Cooper, de Raadt: 2010
  - Mason, Cooper: 2013
  - Mason, Simon: 2016

Studies

- 2001 (de Raadt) - 57 courses
- 2003 (de Raadt) – 85 courses
- Students/course dropped from 349 to 227

- 2010 (Mason) - 44 courses
- Students/course dropped to 176

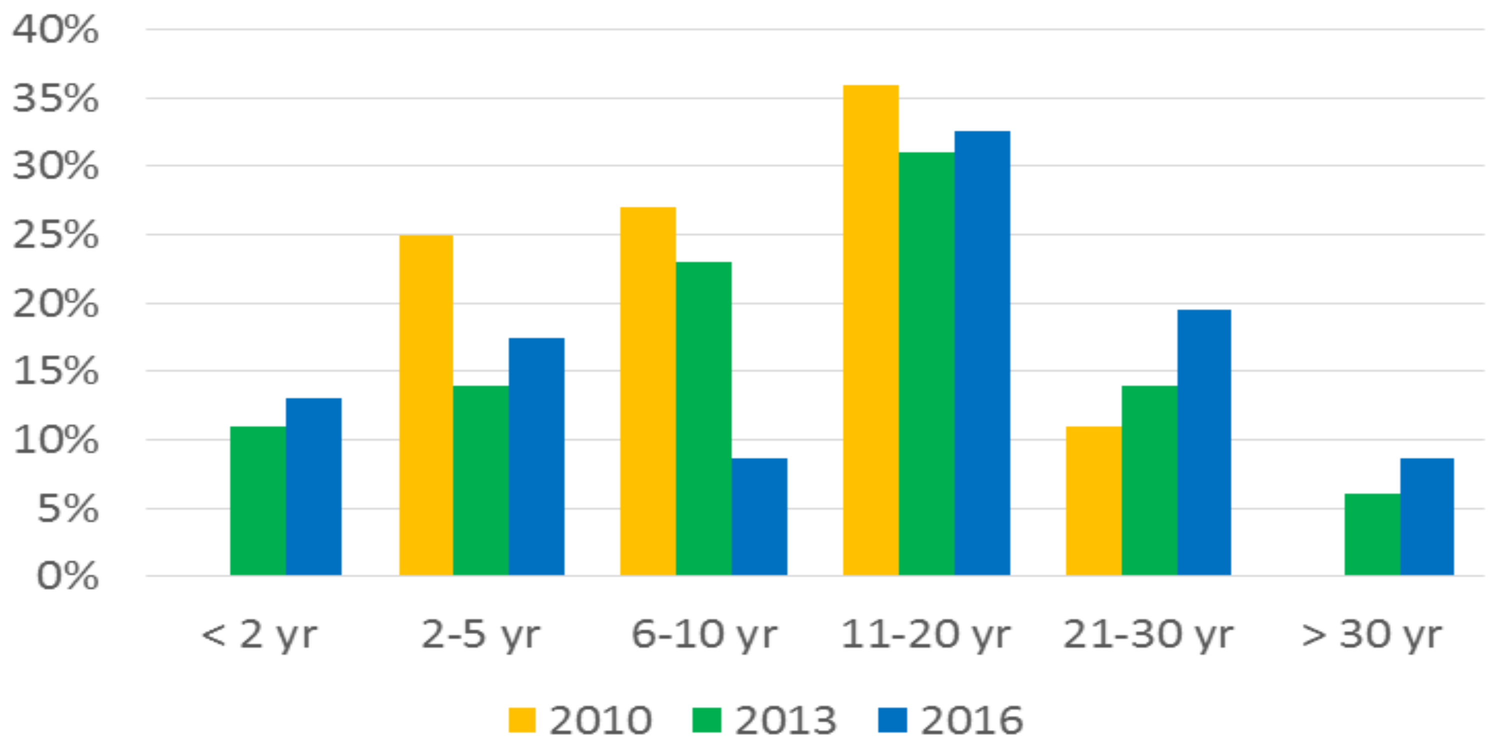
- 2013 (Mason) -   38 courses
- Numbers rising to 264 students/course

2016 Survey demographics

- 48 introductory programming courses
- 35 institutions
- Average course size 423 (a big increase)
- 8 courses < 100, but 8 courses > 1000
- Not all aimed at computing students: some, for example, at science or engineering students

2016 Survey demographics

- Length of teaching experience is interesting



Language trends

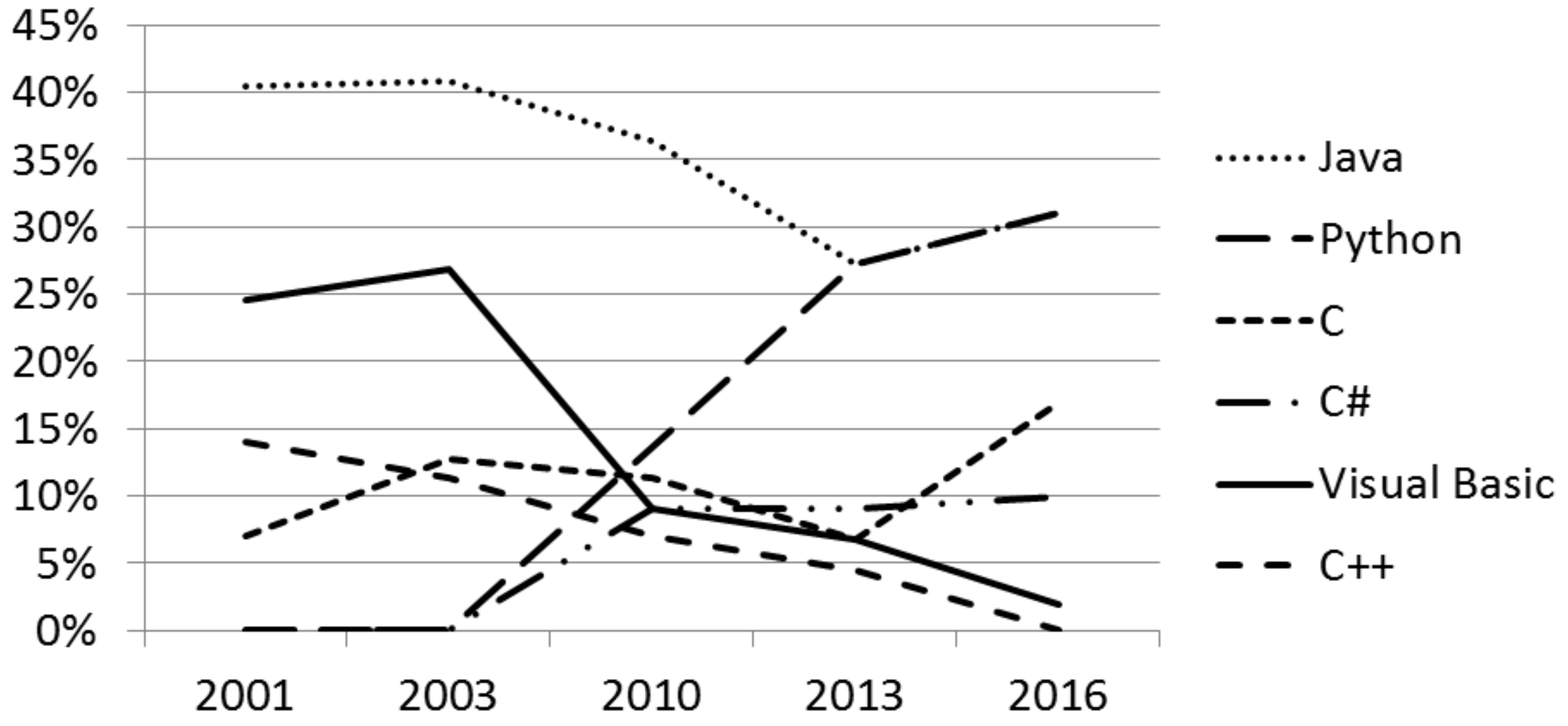
- 2001 – Java (44%) Visual Basic (19%), C++ (15%)
- 2003 – Java (44%), C++ (19%), VB (16%)

- 2010 – Java (39%), Python (20%), C (12%)
- 2013 – Python (34%), Java (27%), Javascript (10%)

- 2016 – Python and Java equal by courses …

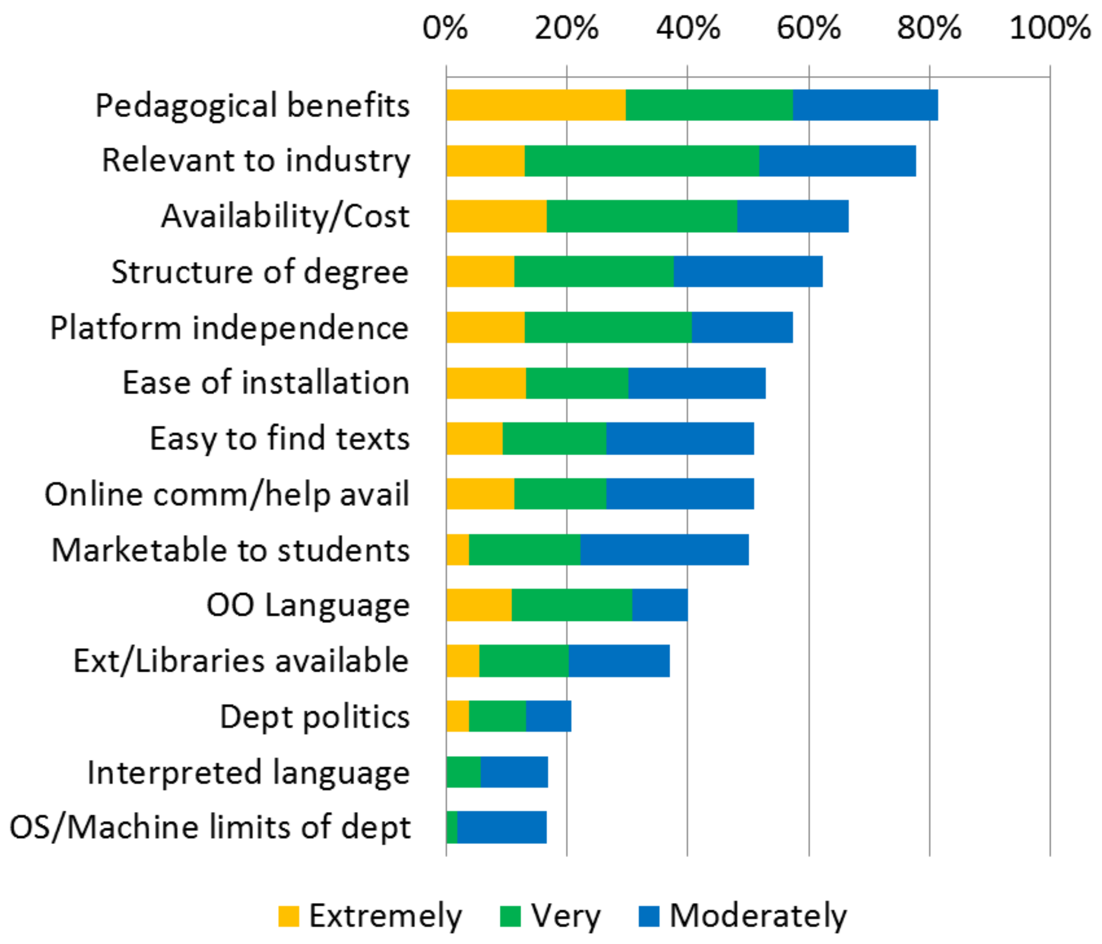
Language trends - 2016

- Python ahead by students (33% vs 28%)
- 14 other languages in use
- Some trends . . .



Reasons for language choice (2016)

- How important was each of these reasons?

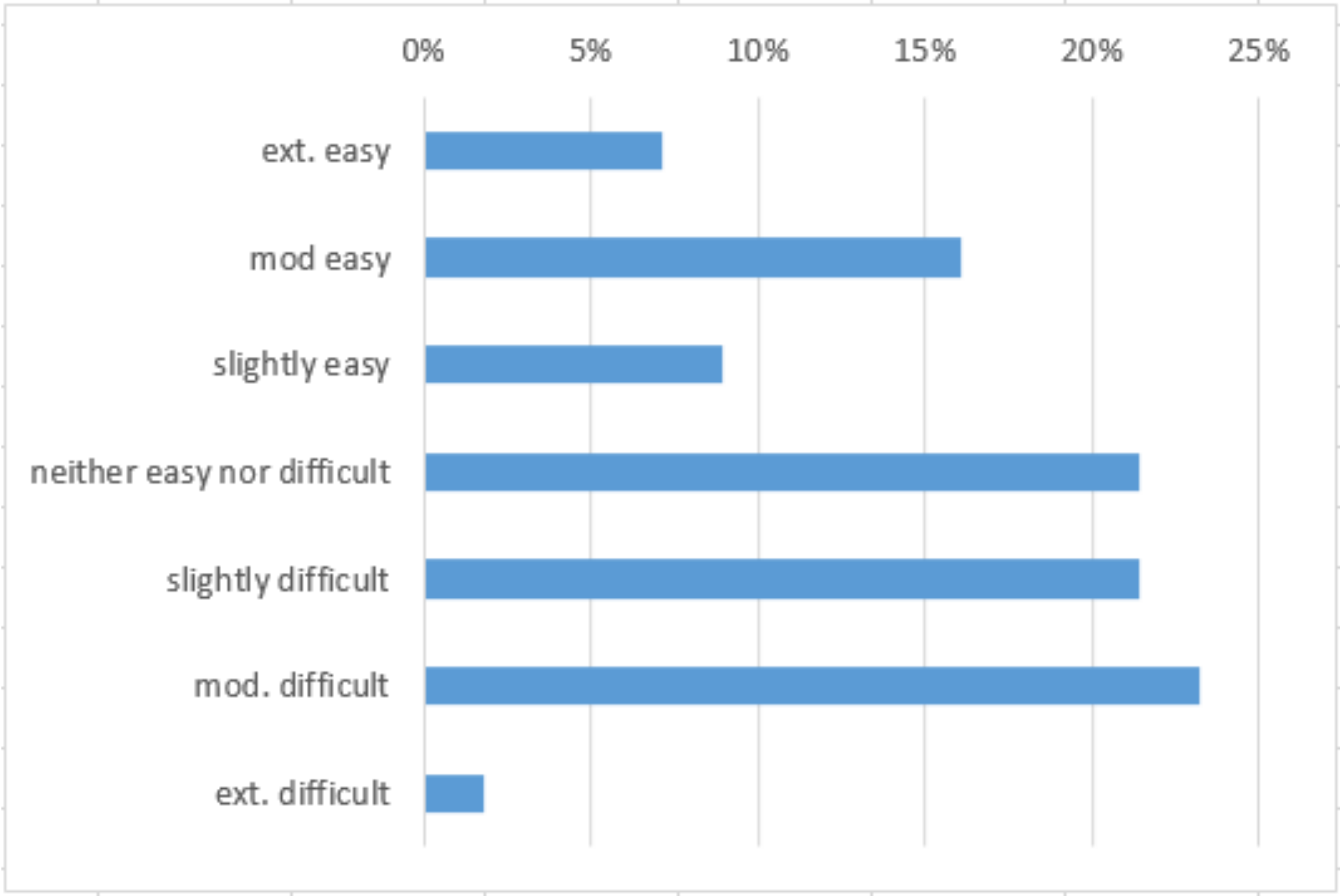


Reasons for choice (Java / Python)?

- Python:
  - Pedagogical benefits (62%)
  - Availability/cost to students
  - Easy to find texts
  - Extensions / Libraries available
  - Platform independence
- Java:
  - Relevance to Industry (92%)
  - OO Language
  - Online community/help available
  - Platform independence

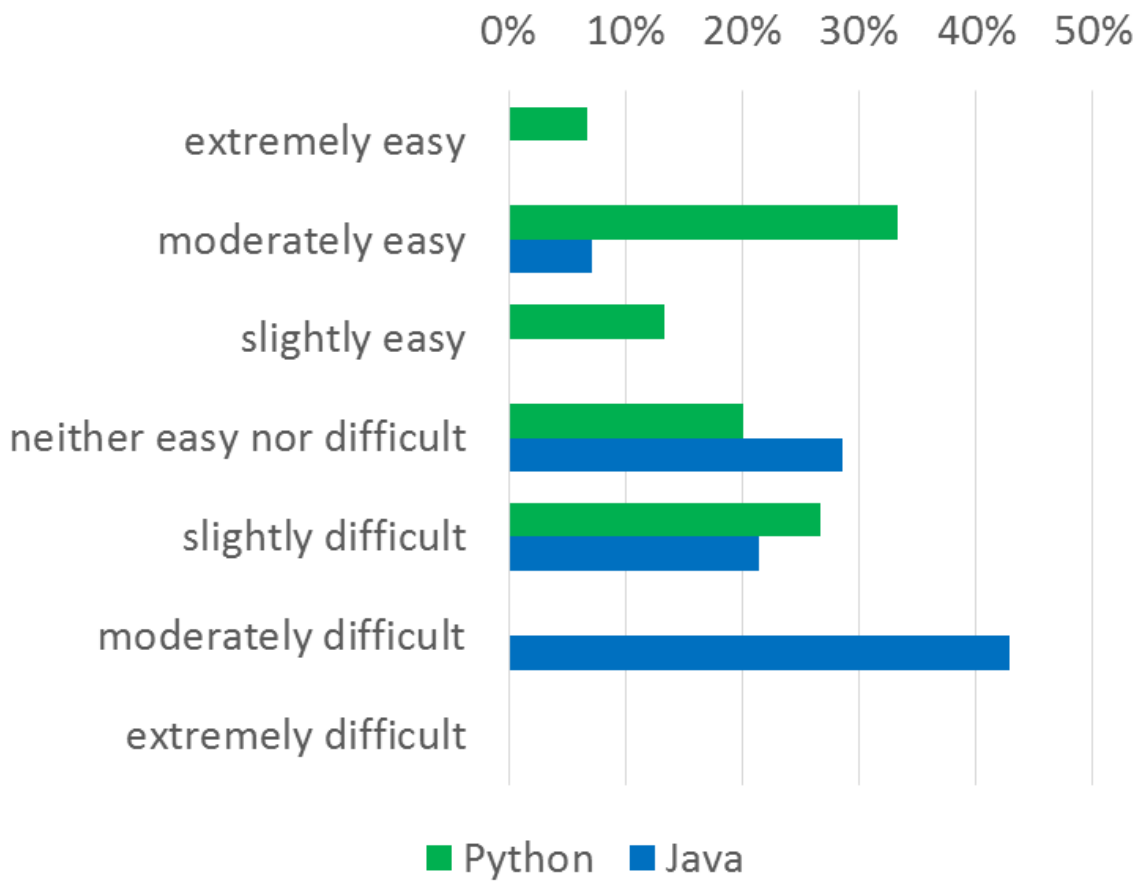
How difficult is the language to learn?

- All languages (2016)



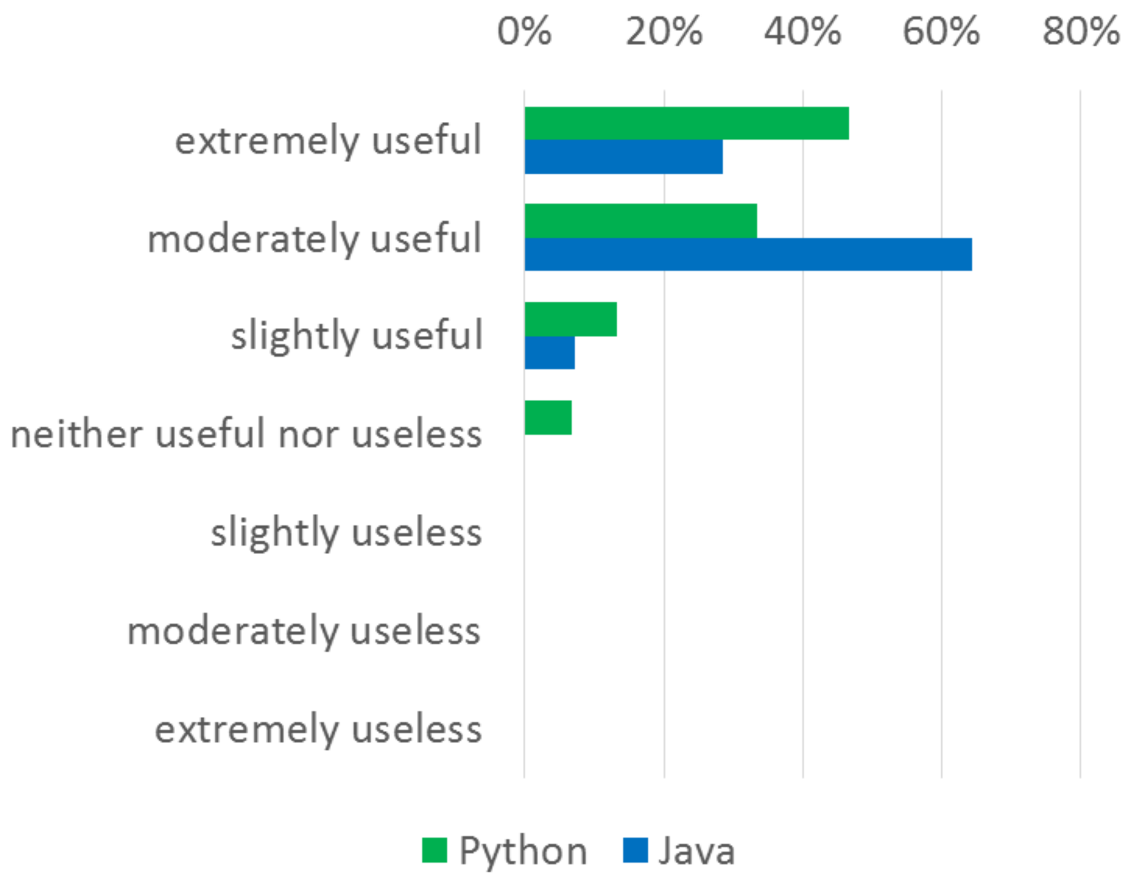
How difficult is the language to learn?

- Broken down just for Java & Python



How useful is the language . . .

- . . . for teaching elementary programming concepts?



Reasons to change language . . .

- Pedagogical Reasons (67%)
- Industry Relevance (42%)
- Degree structure (40%)
- Availability / cost to students (31%)

- "I believe the language used is less important than the teaching pedagogy employed. The language is a tool and effectively any modern programming language could be used to achieve the same end."
- The respondent went on to observe that the principal goal was to overcome students' anxieties about their capabilities, and that one way to achieve this was to provide a programming environment in which students would be comfortable.

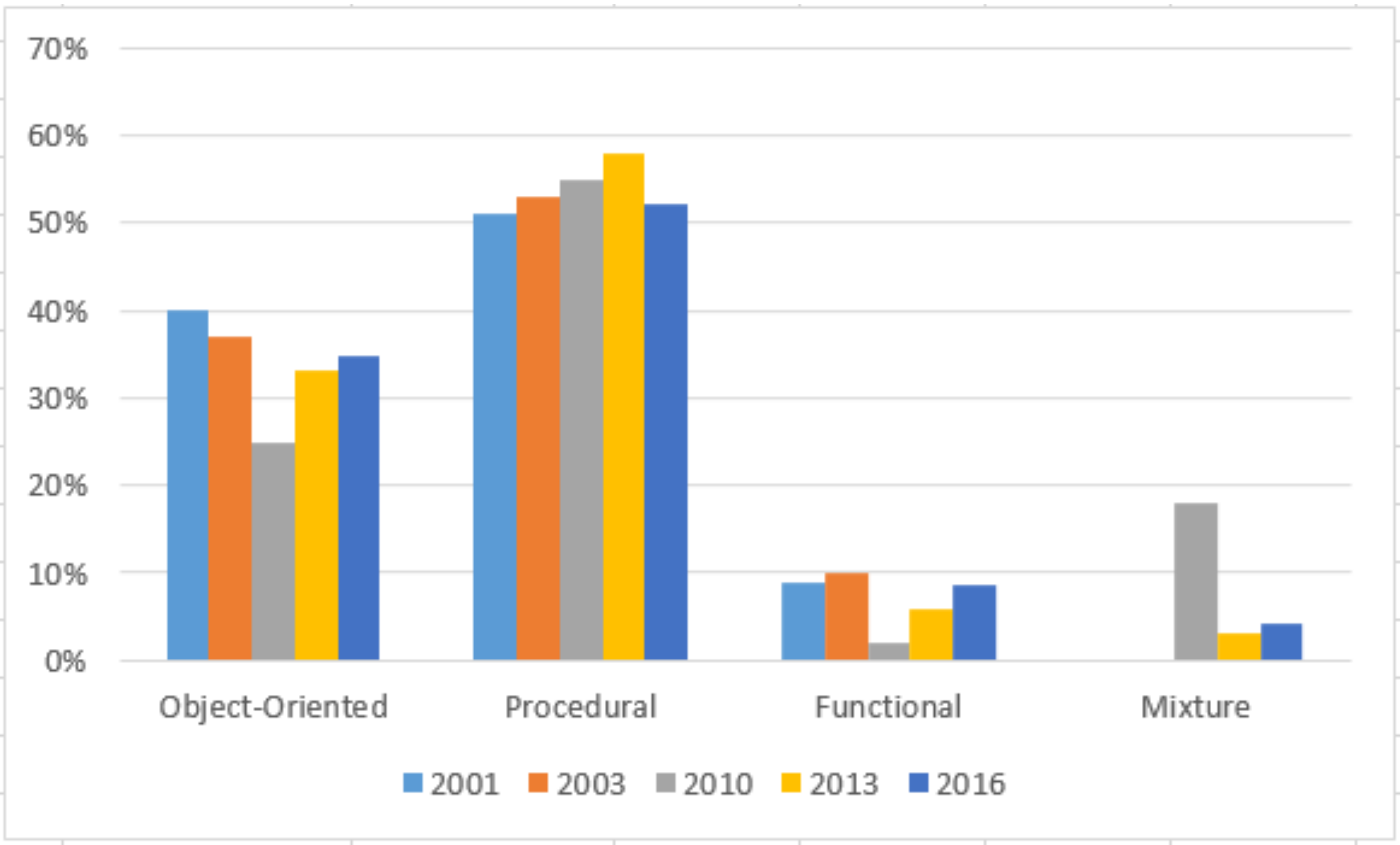
IDEs used with the common languages

Python	IDLE, PyCharm, no IDE, Eclipse, Grok Learning, JES
Java	BlueJ, Eclipse, DrJava, Netbeans, Clara Online, no IDE
C	no IDE, Bloodshed Dev-C++, Visual Studio
Javascript	Brackets
Processing	homegrown IDE, ProcessingJS Gallery
C#	Visual Studio
Visual Basic	Visual Studio, no IDE

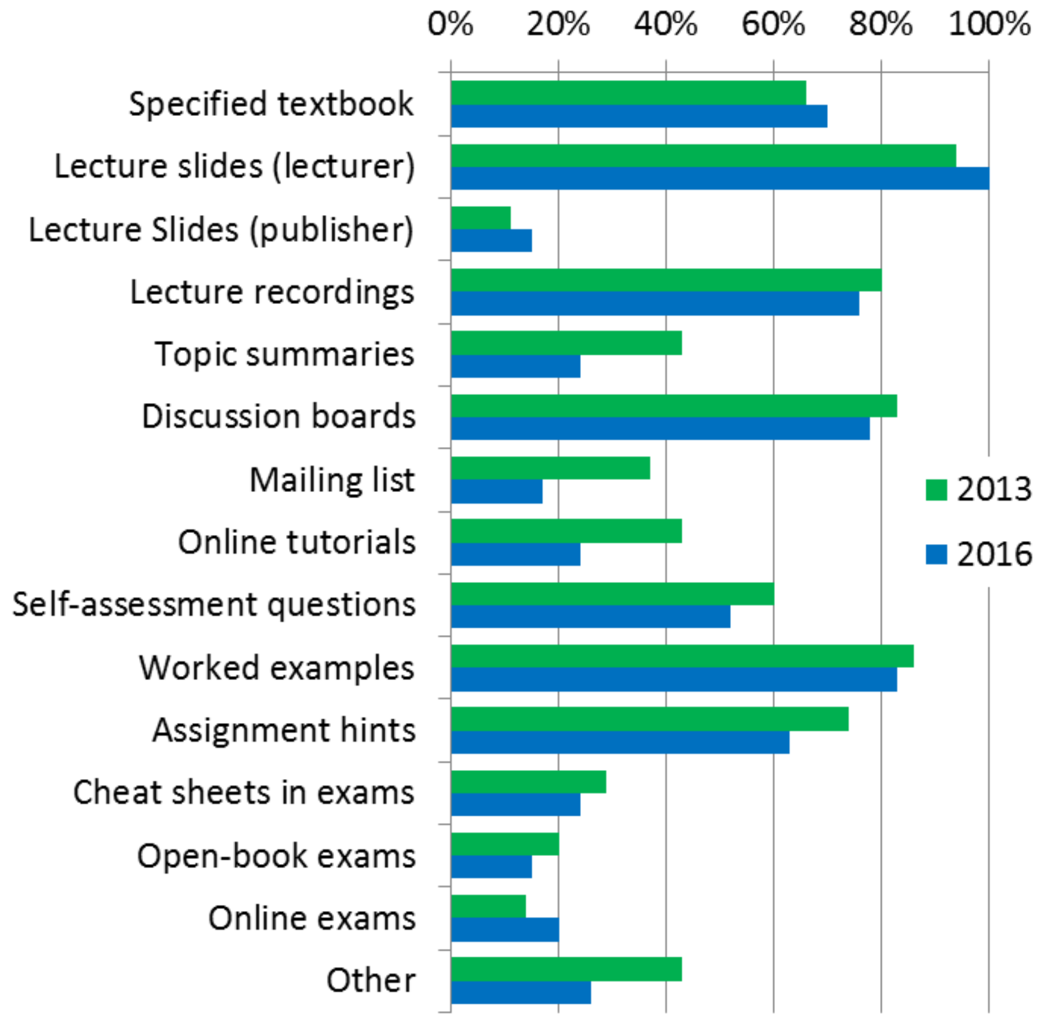
Environments . . .

- Reasons for choice:
  - Uncomplicated / ease of use
  - Pedagogical benefits
  - Availability/cost to students
  - Ease of installation
- Generally:
  - “Learning” IDEs – eg: BlueJ, Alice
  - Industry IDEs – Eclipse, Netbeans, VStudio
  - Rise in cloud-based IDEs
    (Clara’s World, Grok Learning, Brackets)
  - No installation, most cost-free, could see substantial growth in next iteration of study.

Paradigms used . . .



Resources that might help students



So what?

- Why does any of it matter? (Give Peace A Chance)

- We see a rise in the use of Python
  - and a belief that Python is easier to learn
  - and a rise in pedagogical reasons for choosing a language
  - and a large rise in class sizes
  - and increased casualisation of university staff
- Are we struggling to find ways of teaching students form further down the academic scale?

Further research?

- Why are some languages considered ‘easier’? What factors within the language let them be considered “easier”? Or more suitable for teaching?
- “Learning” vs “professional” environments (vs no IDE) – when and where and how
- How are increasing class sizes changing the ways we teach? This research is one piece of the puzzle.