A case for changing the language in CIS*1500

Over the past ten years, the CIS*1500 curriculum has focused largely on the teaching of introductory programming using C as the programming language. Over that time, the class size has more than doubled. However whilst C is a potent programming language, it was never designed as a teaching language. Students often struggle with some of the more inane language syntax, awkward error messages, low-level I/O, and lack of easy-to-use libraries, the language is not one that is easy to learn. Students also tend to struggle with the notion of points and parameter passing in functions. At this point in time, with the use of Raspberry Pi's in the course it may be time to consider migrating to a more learnable programming language.

A new paradigm

In a world where software development is often done in a multi-language environment, it is time to move beyond a single-language introduction to computer science. The purpose of the introductory course in programming is to introduce problem solving, and the elements of programming, which include programming structures, style, testing, and design. The focus should not be on a particular language. There is one main goal to CIS*1500:

- Students learn the basic tenets of programming, and the ability to use a programming language as a tool to solve problems.

Language alternatives

There are a number of alternatives to teaching C, mostly based more on dynamic languages. Whereas traditional programming languages such as C evolved for large-scale programming, scripting languages are designed for simplicity and flexibility and are more amenable to experimentation and incremental development.

- Simple language, simple regular syntax = more time for concepts
- Rich built-ins = interesting programs earlier
- Ease of experimentation
- Emphasize algorithmic thinking

Python is a great language because it is easy to learn, and the emphasis can be placed upon solving problems rather than syntax. Python gets people learning quickly and motivates them into broaching a deeper understanding of programming. Some of the benefits of Python include:

- Portable, dynamically typed scripting language.
- Real-world language - a key language in companies such as Google.
- Dynamic typing - less code, eliminates redeclaration
- Indentation as syntax - promotes and teaches proper code layout
- Interesting projects can be developed with only a fraction of the code that would be required in a system language.

- Able to create programs using graphics.
- Full dynamic run-time type checking and bounds checking on array subscripts.
- Employs garbage collection so there is no problem with dangling pointers or memory leaks.
- Students will spend less time fixing errors.

Another alternative is to use Julia. Julia is a language developed at MIT, and released in 2012. Although designed for scientific programming, Julia offers the learnability of Python, with the speed of C.

Pedagogically Speaking

What does the literature say about teaching languages such as Python or Julia in the CS1 environment? A study was done in 2010 by Enbody and Punch on the impact of changing the CS1 language to Python. They concluded that Python-prepared students performed as well as the non-Python students in spite of the fact that these courses were based on C, and C++.

Their conclusions of the benefits of Python:

- Python has a lower "cognitive load", i.e. the features made available by Python tend to directly help in solving problems. *This coincides well with the principle that an important goal of an introductory programming course should be the ability to use programming as a useful problem-solving tool.*
- With a lower load for learning the language, students can focus more on problem solving.
- Focusing less on new language issues for every problem and focusing more on applying what they already now to a new problem.
- Direct availability of useful data structures and their associated methods.
- It is easier to tackle problems early using real world data.

What about CIS*2500?

- Stays in C.
- Part of class is review of C - spend 1 week going over basic syntax. Concepts like pass-by-reference play into pointers, and can easily be covered.
- The conceptual base is the same
- Switching languages is good for the students

What about CIS*1500 (Winter)?

- More relevant for the majority of students taking the winter version of CIS*1500.

REFs

- Enbody, R.J., Punch, W.F., "Performance of Python CS1 students in mid-level non-Python CS courses", ACM SIGCSE, pp.520-523 (2010)
- Enbody, R.J., Punch, W.F., McCullen, M., "Python CS1 as preparation for C++ CS2", ACM SIGCSE, pp.116-120 (2009)