COM241

Programming Languages Concepts

Assoc. Prof. Dr. Mehmet Serdar Güzel

Slides are mainly adapted from the following course page:
http://www.cs.utexas.edu/~shmat/courses/cs345/

Lecturer

- Instructor: Assoc. Prof Dr. Mehmet S Güzel
  - Office hours: Tuesday, 1:30-2:30pm
  - Open door policy – don't hesitate to stop by!
- Watch the course website
  - Assignments, lab tutorials, lecture notes

Course Materials

- Textbook:

  Mitchell. "Concepts in Programming Languages." (Fifth Edition)

- Harbison, Steele. "C: A Reference Manual."

  (5th edition)

- Occasional assigned readings

Syllabus

- Review of fundamental concepts underlying contemporary programming languages
  - Goal: understand paradigms of programming languages
  - Examples drawn from C, C++, C#, Java, Scheme, Python, PhP
- Procedural
- Functional
- Data-oriented
- Object-oriented
- Script
- Concurrent

Course Goals

- Language as an outline for problem-solving
  - Understand the languages so as to having a fair comparison
  - History of the state-of-the-art programming languages
  - Be prepared for new programming languages, paradigms and tools

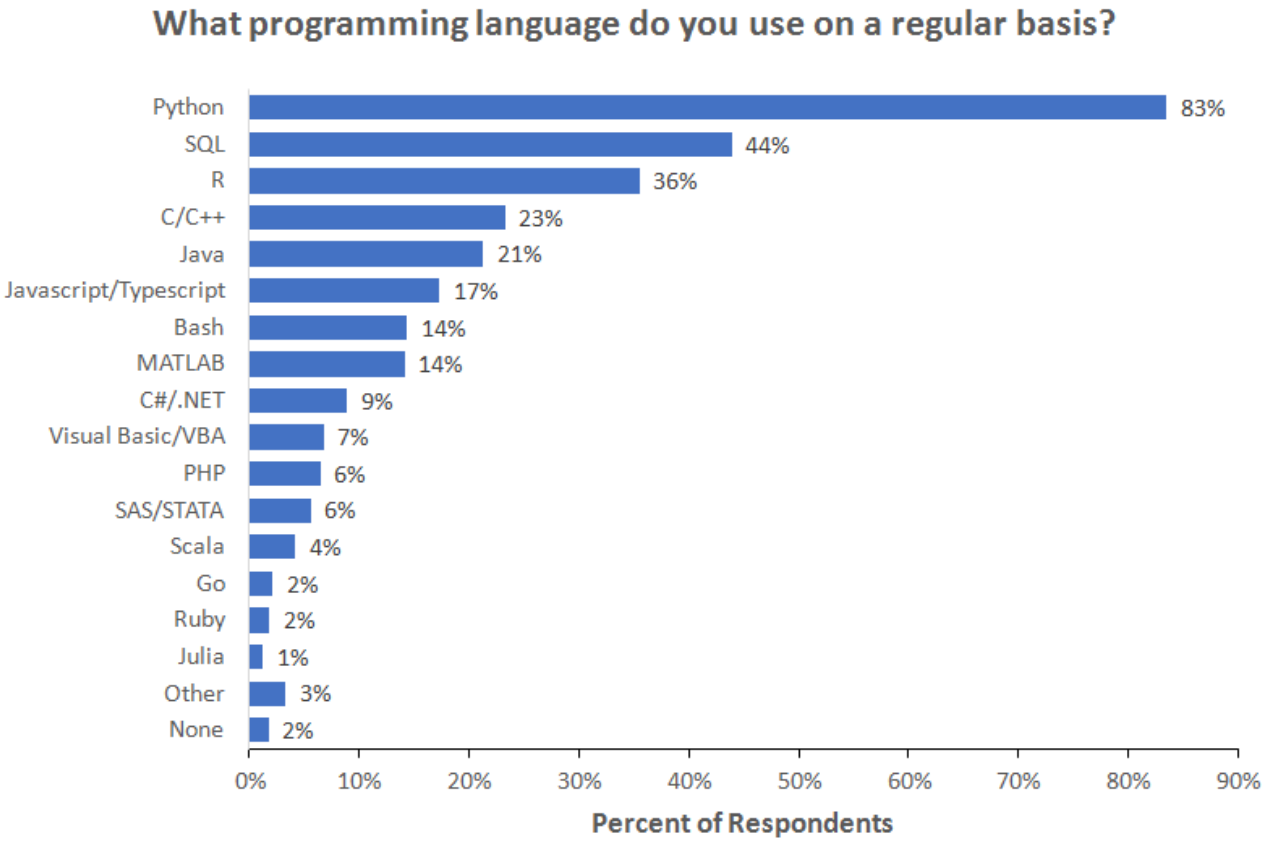
- Language tradeoffs
  - Every suitability has its cost
    - Recognize the cost of presenting an abstract view of machine
    - Understand tradeoffs based on language design

What's Worth Studying?

- Popular languages and standards
  - C, C#, C++, Java and Python
  - Imperative and object-oriented languages
- Important application ideas
- Performance challenges
  - Concurrency
- Design tradeoffs
- Concepts that research community is exploring for new programming languages and tools

Languages in Common Use

[F. Labelle]



Note: Data are from the 2018 Kaggle Machine Learning and Data Science Survey. You can learn more about the study here: http://www.kaggle.com/kaggle/kaggle-survey-2018. A total of 18827 respondents answered the question.

BUSINESS BROADWAY
DATA SCIENCE | CUSTOMER ANALYTICS | MACHINE LEARNING

orge

Flon's Axiom

"There is not now, nor has there ever been,
nor will there ever be,
any programming language in which
it is the least bit difficult to write bad code."

- Lawrence Flon

Latest Issues

- Commercial issues
  - Increasing use of type-safe languages: Java, C#, …
  - Scripting and other languages for Web based applications
  - Scripting languages for AI and Robotics (Python)
- Teaching trends: Java and C# replacing C++
- Research and development issues
  - Modularity
  - Program analysis
    - Automated error detection, programming environments, compilation
  - Isolation and security
    - Sandboxing, language-based security, …

Support for Abstraction

- Data
  - Pre-defined types and classes
  - Class libraries
- Procedural
  - Pre-defined functions
  - Standard libraries

Reliability

- Program behavior is the same on different platforms
  - E.g., early versions of Fortran
- Type errors are detected
  - E.g., Semantic errors are properly trapped
  - E.g., C vs. C++
- Memory leaks are prevented
  - E.g., C# vs. Java
- Pointers are unreliable
  - E.g., C++, C

Orthogonality

- **Orthogonality** in a **programming language** means that a relatively small set of primitive constructs can be combined in a relatively small number of ways to build the control and data structures of the **language**.

Efficient Implementation

- Embedded systems
  - Real-time response (drones)
  - Failures of early Ada implementations
- Web applications
  - Responsiveness to users (e.g., Yandex search)
- Corporate database applications
  - Efficient search and updating
- AI applications
  - Mimicking human behaviors