Introduction

COMP 524: Programming Languages
Srinivas Krishnan
January 11, 2011

Based in part on slides and notes by Bjoern Brandenburg, S. Olivier and A. Block.
Goals

• Gain an understanding of “programming languages”
• Abstractions, Concepts and Implementation Details
• Broaden your knowledge of languages/paradigms

Prerequisites:

➡ COMP 410: Data Structures.
➡ Proficient in Java.
➡ Comfortable with programming.

Expectations

➡ Keep up with the material
➡ Like to learn
➡ Interact !!!
Get to know you

• Broaden your knowledge of languages

What languages do you currently know?

Ever tried learning a new language: C/C++?

Hardest bug you solved

Used a tool: Debugger, Profiler?

Next steps...

Coolest program you wrote...
Motivation

• Gain an understanding of “programming languages”

  Because its Fun !!!

  Strong Fundamentals

  Level 19 Master !!!

  Pickup any new languages
Motivation

• Programming language form the **backbone** of computer science

• Most languages share concepts and abstractions

• Make better use of languages you already know

• Make better and informed decision on picking a language to solve a problem

• Once again It’s fun !!
Trends

Source: http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
Skills

**URI Description using ABNF**

```
ur l
httpaddress | ftpaddress | newsaddress | nntpaddress | prosperoaddress | telnetaddress | gopheraddress | waisaddress | mailtoaddress | midaddress | cidaddress
scheme
ialpha
httpaddress
http : // hostport [ / path ] [ ? search ]
ftpaddress
ftp : // login / path [ ; ftptype ]
```

- Detail specifications often in **BNF**
- Know how to parse such documents like RFC’s
Topics/Scope

Foundations.
- Syntax and syntactical analysis.
- Binding, scope, and storage.
- Semantic analysis.

Paradigms.
- Object orientation.
- Functional programming.
- Logic programming.
- Scripting languages.

Core language design.
- Control flow and subroutines.
- Evaluation strategies.

Select high-impact topics.
- Concurrency.
- Security concerns.
- Runtime systems.

Programming languages.
- Java.
- Prolog.
- LISP
- Python.
Topics/Scope

Foundations.
- Syntax and syntactical analysis.
- Binding, scope, and storage.
- Semantic analysis.

Paradigms.
- Object orientation.
- Functional programming.
- Logic programming.
- Scripting languages.

Core language design.
- Control flow and subroutines.
- Evaluation strategies.

Select high-impact topics.
- Concurrency.
- Security concerns.
- Runtime systems.

Programming languages.
- Java.
- Prolog.
- Haskell.
- Python.

Notable omissions.
- Formal background (see COMP 455).
- Target architectures (see COMP 411).
- Code generation and optimization (covered in COMP 520).
- Formal treatment of semantics (advanced grad level topic of little practical relevance).
Class Rules

Let’s have a look at the syllabus…