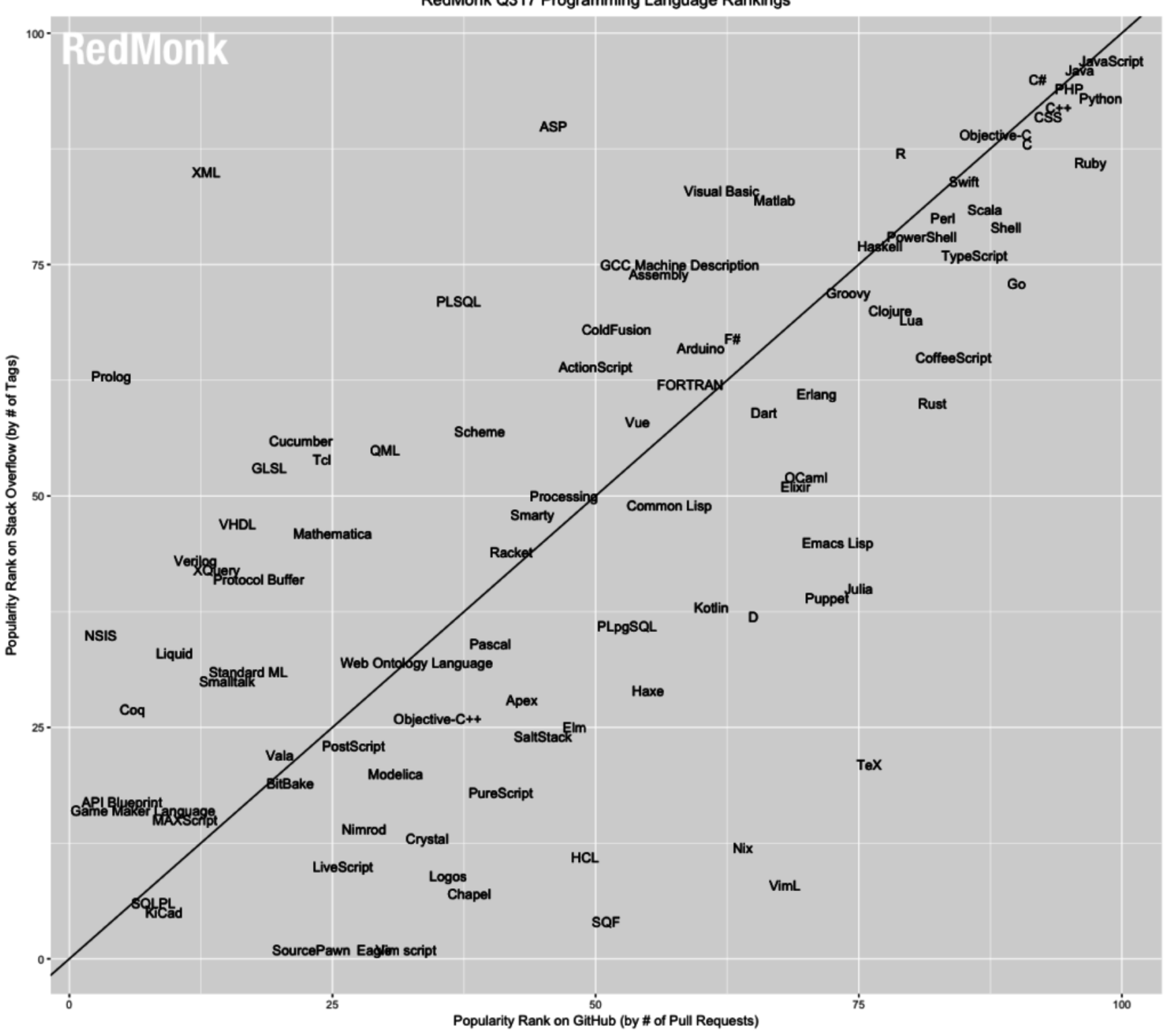
PL 2017 Winter

Programming Languages & Algorithms

Provided by **Tae Geun Kim**

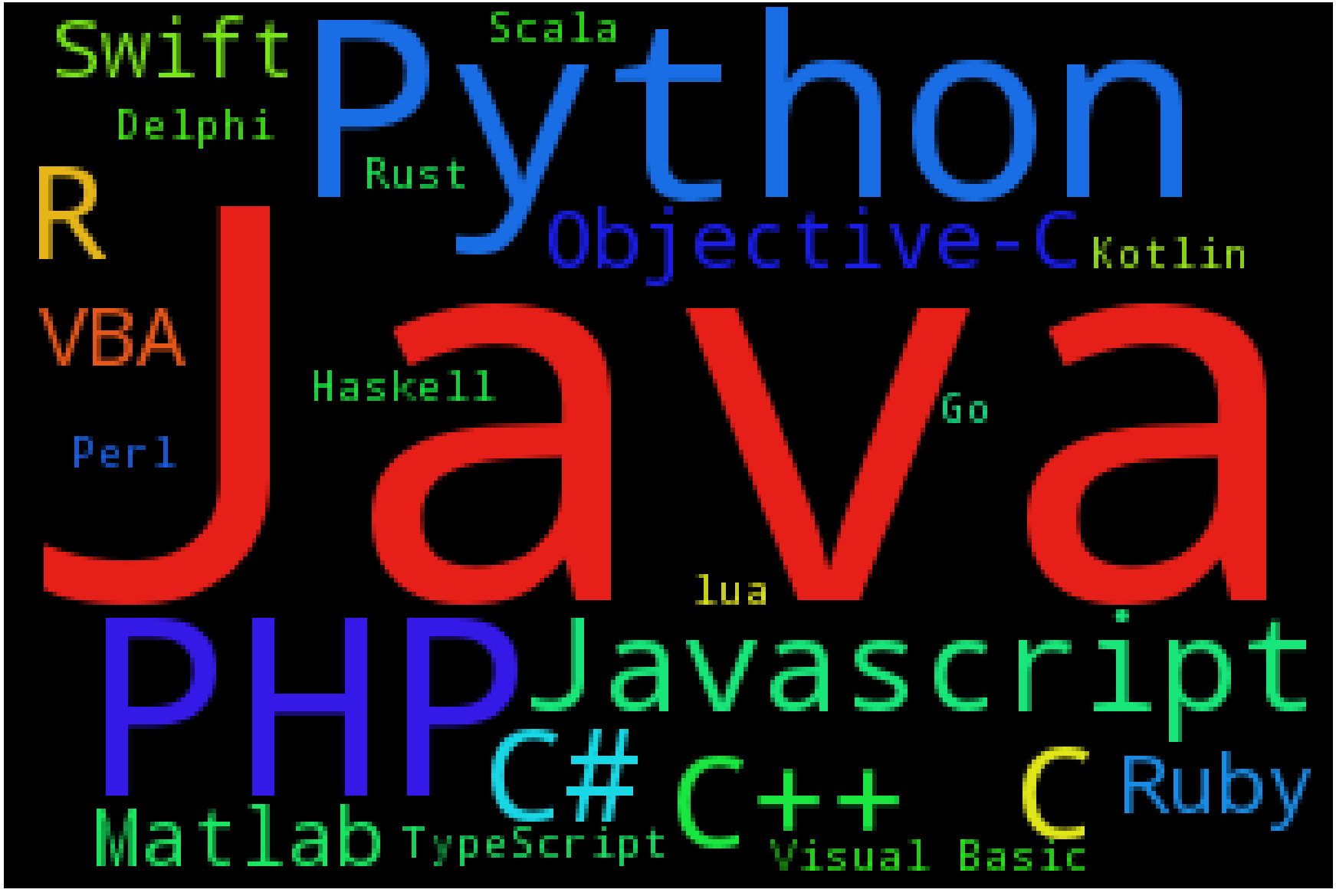
Programming Languages

- What's the most popular programming languages?
  - TIOBE Index
  - Redmonk
  - Slant
  - PYPL


RedMonk Q317 Programming Language Rankings
RedMonk
Popularity Rank on Stack Overflow (by # of Tags)
Popularity Rank on GitHub (by # of Pull Requests)
0
25
50
75
100
JavaScript
Java
C#
PHP
Python
C++
CSS
Objective-C
C
R
Ruby
Swift
Visual Basic
Matlab
Scala
Perl
Shell
PowerShell
Haskell
TypeScript
GCC Machine Description
Assembly
Go
Groovy
Clojure
Lua
PLSQL
ColdFusion
F#
Arduino
CoffeeScript
ActionScript
Prolog
FORTRAN
Erlang
Rust
Dart
Vue
Scheme
Cucumber
QML
Tcl
GLSL
OCaml
Elixir
Processing
Common Lisp
Smarty
VHDL
Mathematica
Emacs Lisp
Racket
Verilog
XQuery
Protocol Buffer
Julia
Puppet
Kotlin
D
PLpgSQL
NSIS
Pascal
Liquid
Web Ontology Language
Standard ML
Smalltalk
Haxe
Apex
Coq
Objective-C++
Elm
SaltStack
Vala
PostScript
TeX
Modelica
BitBake
PureScript
API Blueprint
Game Maker Language
MAXScript
Nimrod
Crystal
Nix
HCL
LiveScript
Logos
VimL
Chapel
SQL
PL
KiCad
SQF
SourcePawn
Eagle
Vim script
XML
ASP

Classification - 1. Uses

- **Major Languages** - Java, JS, C/C++, C#, Python
- **Grow Languages** - Swift, Scala, Go, Rust, F#, Kotlin
- **DSL** - R, Matlab, TeX, Mathematica, SQL
- **Dead** - FORTRAN, Scheme, LISP, Objective-C
- **Base** - Assembly


Swift
Scala
Python
Delphi
Rust
R
Objective-C
Kotlin
VBA
Java
Haskell
Go
Perl
lua
PHP
Javascript
C#
C++
C
Ruby
Matlab
TypeScript
Visual Basic

Classification - 2. Type

- **Procedural -** C, D, FORTRAN, ALGOL
- **Object-Oriented -** Java, C++, C#, Go, Rust
- **Functional -** Haskell, Elixir, Racket, Clojure
- **Multi-Paradigm -** Scala, F#, Julia, Swift, Python

Programming Pradigm

- 명령형 프로그래밍(imperative programming)

> 명령형 프로그래밍은 문제를 해결하는 절차를 기술하는 방식의 프로그램의 스타일. 프로그램은 수행할 명령어들로 구성. 명령어들은 주로 프로그램의 상태를 변경함

- 함수형 프로그래밍(functional programming)

> 프로그램의 계산 과정을 수학 함수의 수행으로 간주하는 프로그래밍 스타일. 프로그램은 함수의 정의들로 구성됨. 함수 수행은 부수 효과를 허용하지 않는다. (함수의 부수효과란 상태변경, 데이터 수정을 의미한다)

- 논리 프로그래밍(logic programming)

  > 정형 논리를 기반으로 한 프로그래밍 스타일. 프로그램은 문제에 대한 사실 혹은 규칙을 표현하는 논리 문장들의 집합.

- 객체 지향 프로그래밍(object-oriented programming)

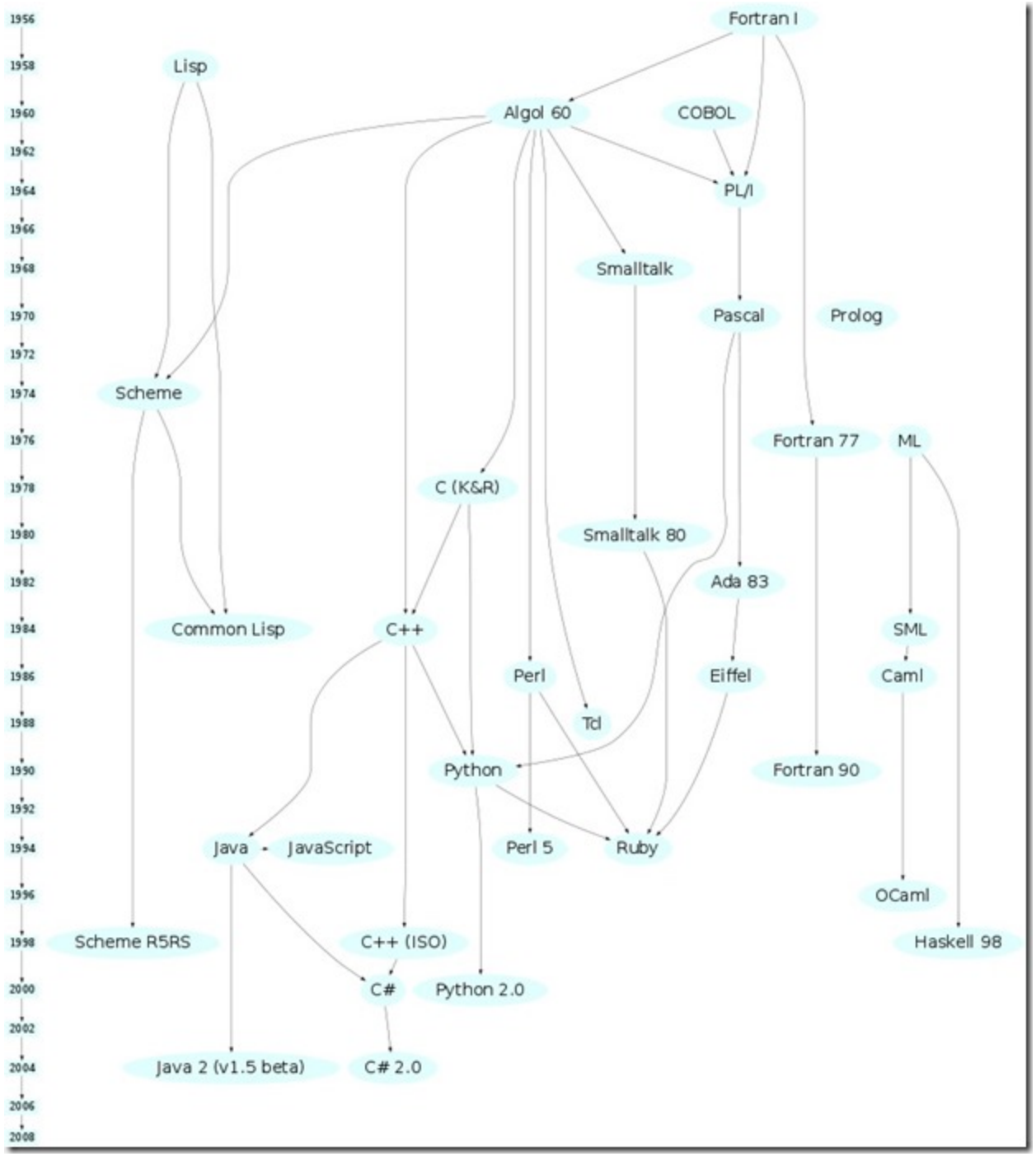
  > 객체개념을 기반으로 하는 프로그래밍 스타일. 객체는 data와 그에대한 연산(프로시저,메소드)들을 포괄하는 개념. 프로그램의 실행은 객체사이의 상호작용에 의해 이뤄짐.

Paradigm Reference

- Paradigm Introduction
- Paradigm Classifications
- Imperative vs Functional

Classification - 3. History

- **Ancienct -** FORTRAN, ALGOL, Cobol, LISP
- **Old -** C, Pascal, Prolog
- **Proper -** C++, Java, Python, Erlang, Haskell
- **Young -** C#, Go, Swift, Rust, Kotlin, Scala, Julia


1956
1958
1960
1962
1964
1966
1968
1970
1972
1974
1976
1978
1980
1982
1984
1986
1988
1990
1992
1994
1996
1998
2000
2002
2004
2006
2008
Fortran I
Lisp
Algol 60
COBOL
PL/I
Smalltalk
Pascal
Prolog
Scheme
Fortran 77
ML
C (K&R)
Smalltalk 80
Ada 83
Common Lisp
C++
SML
Perl
Eiffel
Caml
Tcl
Python
Fortran 90
Java
JavaScript
Perl 5
Ruby
OCaml
Scheme R5RS
C++ (ISO)
Haskell 98
C#
Python 2.0
Java 2 (v1.5 beta)
C# 2.0

History of Programming Language

1943 - ENIAC coding system
1951 - Regional Assembly Language
1952 - Autocode
1954 - IPL (forerunner to LISP)
1955 - FLOW-MATIC (forerunner to COBOL)
1957 - FORTRAN (First compiler)
1957 - COMTRAN
1958 - LISP
1958 - ALGOL 58
1959 - FACT
1959 - COBOL
1959 - RPG
1962 - APL
1962 - Simula
1962 - SNOBOL
1963 - CPL (forerunner to C)
1964 - BASIC
1964 - PL/1
1967 - BCPL (forerunner to C)
1968 - Logo
1969 - B (forerunner to C)
1970 - Pascal
1970 - Forth
1972 - C
1972 - Smalltalk
1972 - Prolog
1973 - ML
1975 - Scheme
1978 - SQL
1980 - C++
1983 - Ada
1984 - Common Lisp
1984 - MATLAB
1985 - Eiffel
1986 - Objective-C
1986 - Erlang
1987 - Perl
1988 - Tcl
1988 - Mathematica
1989 - FL (Backus);
1990 – Haskell
1991 – Python
1991 - Visual Basic
1991 - HTML
1993 - Ruby
1993 - Lua
1994 - CLOS (part of ANSI Common Lisp)
1995 - Java
1995 - Delphi (Object Pascal)
1995 - JavaScript
1995 - PHP
1996 - WebDNA
1997 - Rebol
1999 - D
2000 - ActionScript
2001 - C#
2001 - Visual Basic .NET
2002 - F#
2003 - Groovy
2003 - Scala
2003 - Factor
2007 - Clojure
2009 - Go
2011 - Dart

For Data Science

- Performance
- Statistics Package
- Visualization
- Readability

Best Performance

1. FORTRAN
2. C++
3. Rust
4. Chapel
5. C#
6. Go
7. Swift
8. Julia
9. Scala
10. Java

Worst Performance

1. Octave
2. R
3. Mathematica
4. Python
5. MATLAB

Performance Reference

- NASA
- Computer Benchmark Game
- Julia Benchmark

Statistical Packages

1. R
2. MATLAB
3. Mathematica
4. Python
5. Julia
6. Scala

Visualization

1. Python
2. Mathematica
3. MATLAB
4. R
5. Julia
6. Scala (Spark, Java)
7. C++ (CERN-ROOT)

Recommended Languages

- DSL - Julia, R
- OOP - C#(Class), Go(Type & Method)
- FP - Haskell(Pure), Scala(FP+OOP), Swift(Impure)
- Web - Python(Django, Flask), Spring, JS
- High Performance - C++

> For Data Scientist,
>
> - Main : R, Julia, Python, Scala
> - Sub : Go, Swift, C++

Homework #1

Submit the report for one language of belows:

- Go
- Swift
- Kotlin
- Haskell

Report should contain belows & should be written in Markdown

- History & Purpose
- Advantange & Disadvantage
- Example Code