



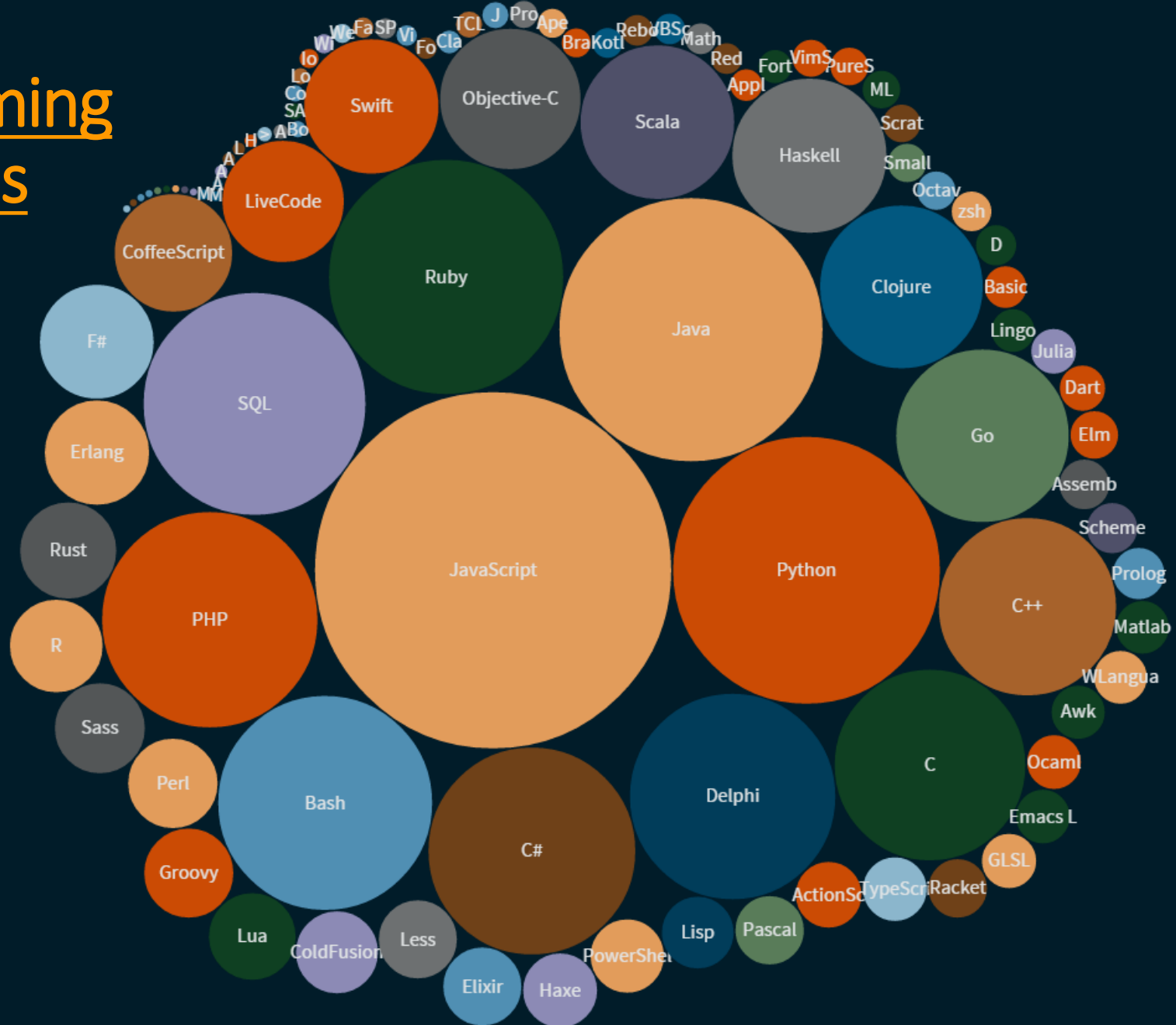
Programming Languages Pragmatics

INTRODUCTION

DR. ERIC CHOU

IEEE SENIOR MEMBER

Programming Languages





Machine Code

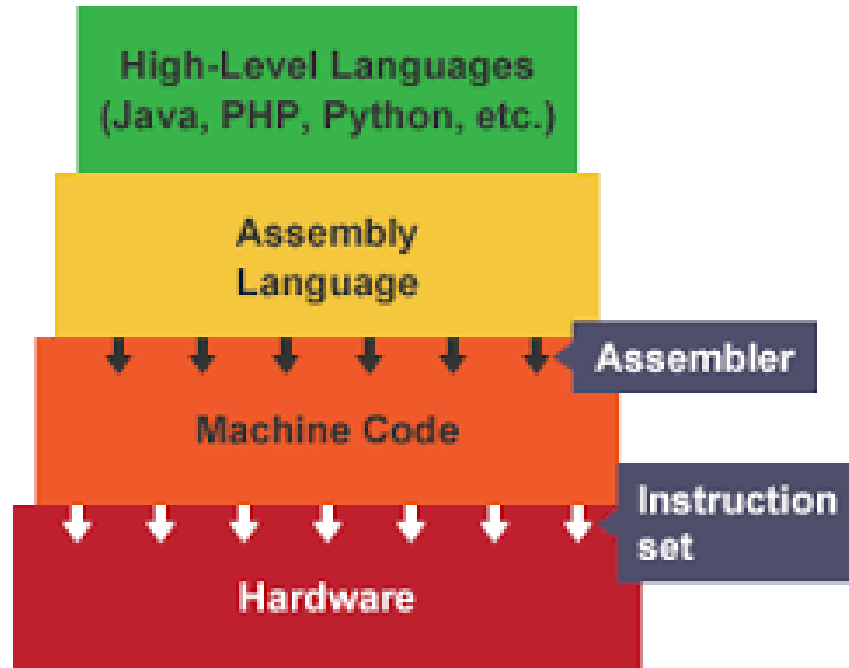
Unreadable

b8 00 b8 8e c0 8d 36 30 03 e8 fd 01 bf a2 00 b96
02 00 eb 2b b4 06 b2 ff cd 21 3c 71 0f 84 a5 01	...+.....l%q....
3e 50 b9 a0 00 74 18 3e 48 b9 a0 00 0f 84 d9 00	<@....6.<H.....
b9 02 00 3e 4d 74 08 3e 4b 0f 84 cc 00 eb d5 89	...<Hc.<K.....
3e b5 09 01 cf 89 3e b3 09 e8 87 01 8b 3e b5 09	>.....>.....>..
b0 20 26 88 06 26 88 45 fe 26 88 86 62 ff 26 88	. 6...6.E.6...b.6.
86 60 ff 26 88 86 5e ff 26 88 86 9e 00 b0 07 26	. ^..6...^..6.....6
88 45 01 8b 3e b3 09 89 fb 83 eb 02 d1 fb 8a 00	.K..b.....
26 88 45 fe 89 fb 81 eb a2 00 d1 fb 8a 00 26 88	6.E.....6.
86 5e ff 89 fb 81 eb a0 00 d1 fb 8a 00 26 88 86	.^.....6..
60 ff 89 fb 81 eb 9e 00 d1 fb 8a 00 26 88 86 62	.^.....6...b
ff 89 fb 81 eb a2 00 d1 fb 8a 00 26 88 86 8e ff6...^.
89 fb 88 c3 02 d1 fb 8a 00 26 88 45 02 89 fb 816.E....
c3 2e 00 d1 fb 8a 00 26 88 86 9e 00 89 fb 81 c36.....
a0 00 d1 fb 8a 00 26 88 86 a0 00 89 fb 81 c3 a26.....
00 d1 fb 8a 00 26 88 86 a2 00 b0 03 26 88 06 a06.....6...
b7 09 26 88 45 01 e9 0b ff 89 3e b5 09 29 cf 89	..6.E.....>..>..
3e b3 09 e8 bd 00 8b 3e b5 09 b0 20 26 88 06 26	>.....>.... 6...6
88 45 02 26 88 86 9e 00 26 88 86 a0 00 26 88 86	.E.6....6....6...
a2 00 26 88 86 62 ff b0 07 26 88 45 01 8b 3e b3	..6...b...6.E...>.
09 89 fb 83 eb 02 d1 fb 8a 00 26 88 45 fe 89 fb6.E...
81 eb a2 00 d1 fb 8a 00 26 88 86 5e ff 89 fb 816...^....
eb a0 00 d1 fb 8a 00 26 88 86 60 ff 89 fb 81 eb6...^.....
9e 00 d1 fb 8a 00 26 88 86 62 ff 89 fb 81 eb a26...b.....
00 d1 fb 8a 00 26 88 86 8e ff 89 fb 83 c3 02 d16...^.....



Assembly Language

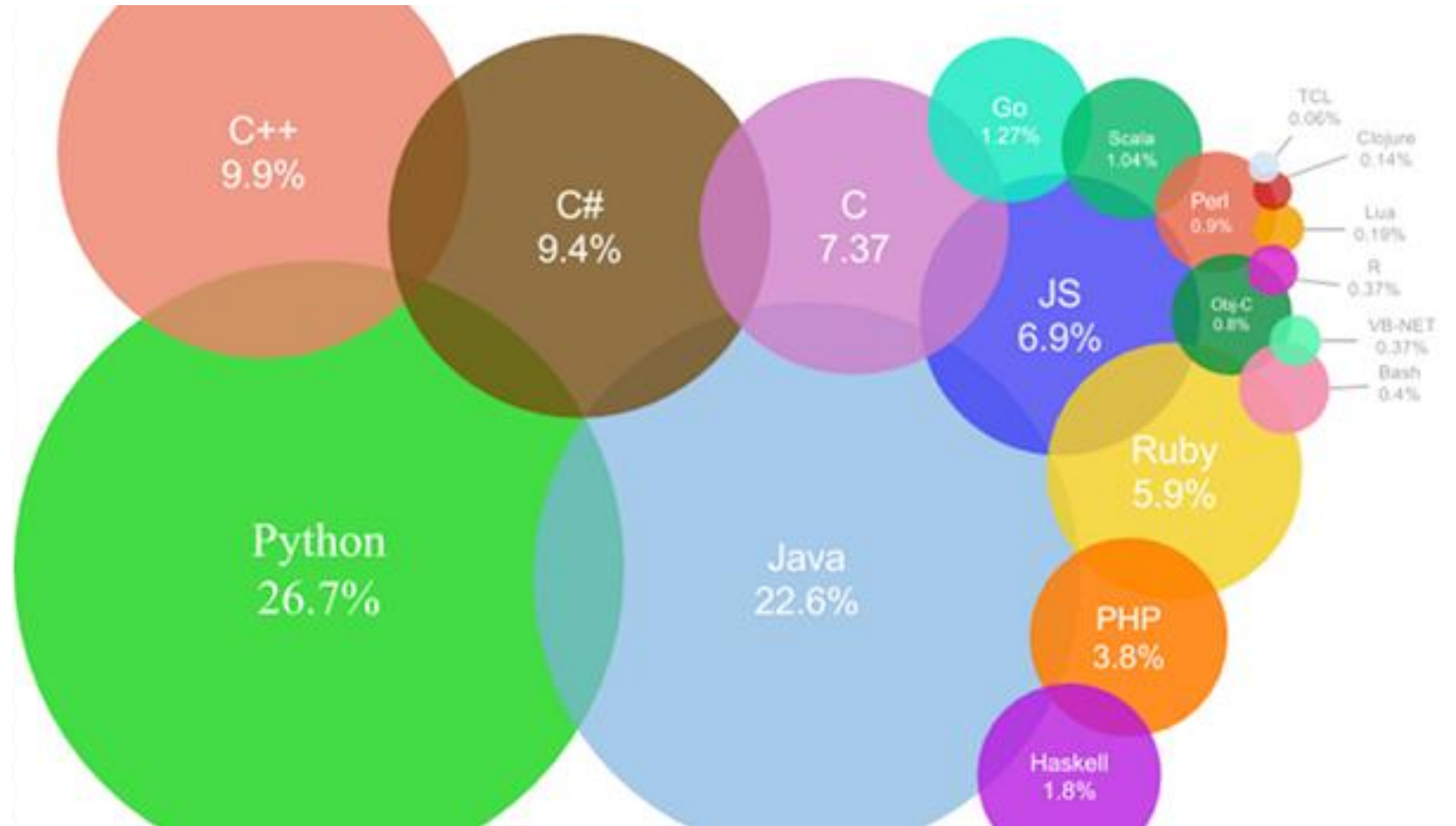
GCD Code



```
pushl    %ebp                # \
movl     %esp, %ebp          # ) reserve space for local variables
subl     $16, %esp           # /
call     getint               # read
movl     %eax, -8(%ebp)       # store i
call     getint               # read
movl     %eax, -12(%ebp)      # store j
A: movl   -8(%ebp), %edi       # load i
movl     -12(%ebp), %ebx      # load j
cmpl     %ebx, %edi           # compare
je        D                   # jump if i == j
movl     -8(%ebp), %edi       # load i
movl     -12(%ebp), %ebx      # load j
cmpl     %ebx, %edi           # compare
jle       B                   # jump if i < j
movl     -8(%ebp), %edi       # load i
movl     -12(%ebp), %ebx      # load j
subl     %ebx, %edi           # i = i - j
movl     %edi, -8(%ebp)       # store i
jmp       C
B: movl   -12(%ebp), %edi      # load j
movl     -8(%ebp), %ebx       # load i
subl     %ebx, %edi           # j = j - i
movl     %edi, -12(%ebp)      # store j
C: jmp    A
D: movl   -8(%ebp), %ebx       # load i
push     %ebx                 # push i (pass to putint)
call     putint               # write
addl     $4, %esp             # pop i
leave    # deallocate space for local variables
mov      $0, %eax             # exit status for program
ret                                # return to operating system
```

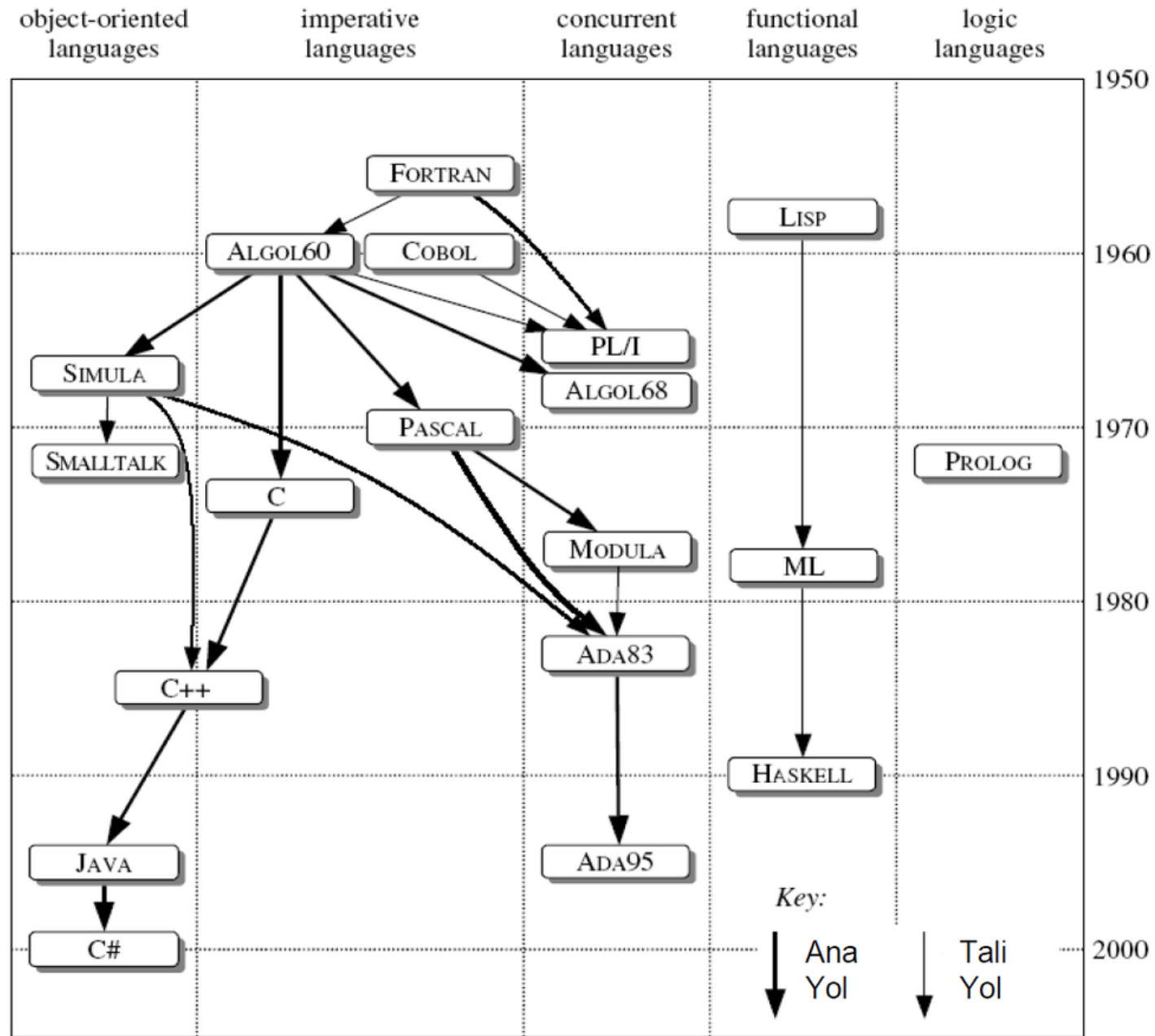


Most Popular Programming Languages 2016





Timeline





Why are there so many programming languages?

1. evolution -- we've learned better ways of doing things over time
2. socio-economic factors: proprietary interests, commercial advantage
3. orientation toward special purposes
4. orientation toward special hardware
5. http://cdn.oreillystatic.com/news/graphics/prog_lang_poster.pdf



What makes a language successful?

1. easy to learn (BASIC, Pascal, LOGO, Scheme)
2. easy to express things, easy use once fluent, "powerful" (C, Common Lisp, APL, Algol-68, Perl)
3. easy to implement (BASIC, Forth, Python)
4. possible to compile to very good (fast/small) code (Fortran, C)
5. backing of a powerful sponsor (COBOL, PL/1, Ada, Visual Basic, C#)
6. wide dissemination at minimal cost (Pascal, Turing, Java, Javascript)



Why do we have programming languages? What is a language for?

1. way of thinking -- way of expressing algorithms
2. languages from the user's point of view
3. abstraction of virtual machine -- way of specifying what you want
4. the hardware to do without getting down into the bits
5. languages from the implementor's point of view



The Art of Language Design

Evolution

Ease of Implementation

Special Purposes

Standardization

Personal Preference

Open Source

Expressive Power

Excellent Compiler

Ease of Use for Novice

Economics/Patronage/Inertia